basic system

Gene Regulation
Experiments

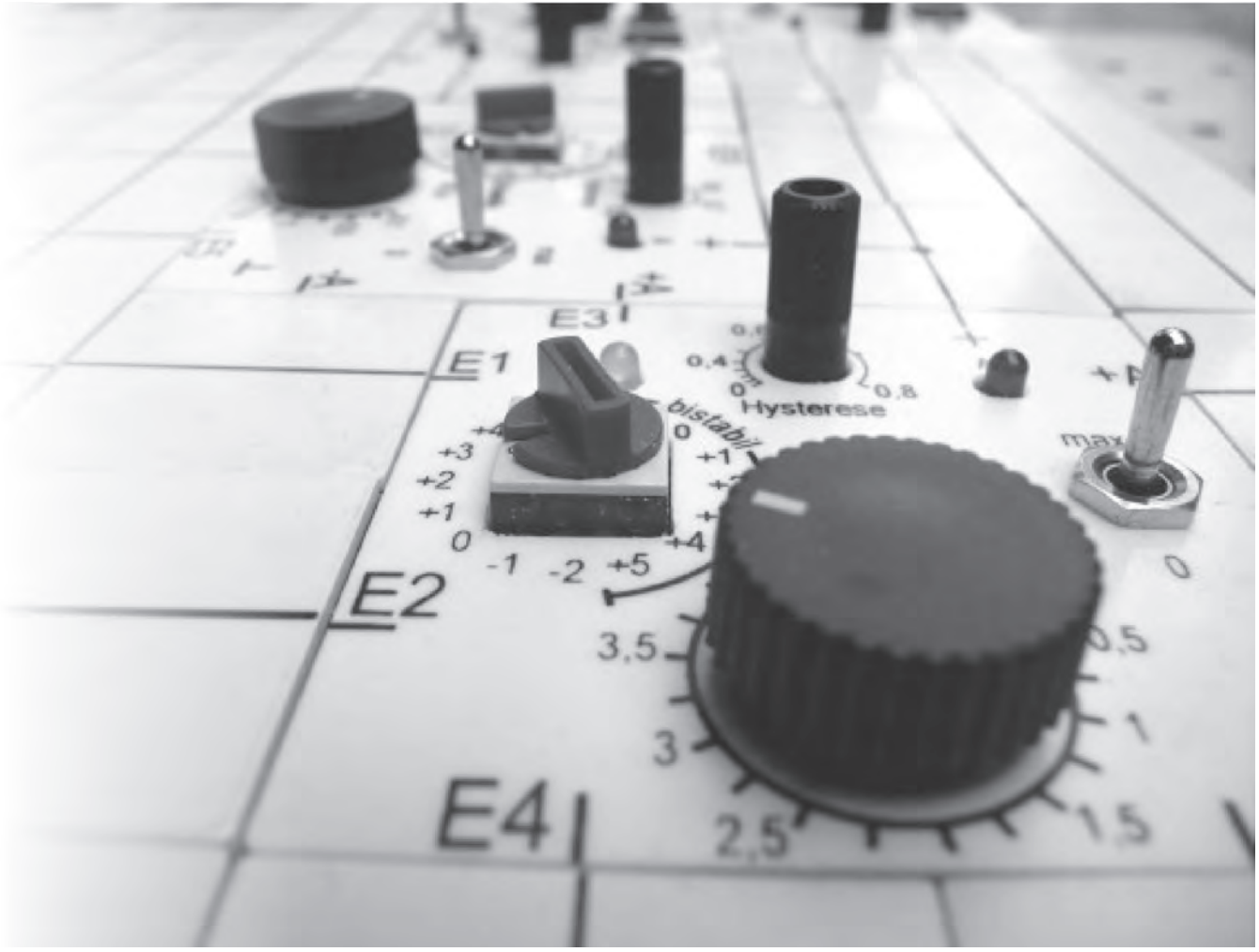
Lectron

The cover shows the flower of the wall-cress (*Arabidopsis thaliana*) in front of a detail of its gene map. The life cycle of this plant lasts only 8 weeks. As Arabidopsis is easy to cultivate it has become the model plant of botanists and geneticists. Its genome with about 25 000 genes was decoded in the year 2000; the operating mode is still not known in large parts, though. A small, meanwhile understood part controls the development of its petals. You can set up and simulate the corresponding network of its gene-connections with this LECTRON experimental kit (experiment 31).

LECTRON Experiments
Gene Regulation
Authors
Dr. Stefan Bornholdt
Gerd Kopperschmidt

Lectron

Instructions
Gene Regulation



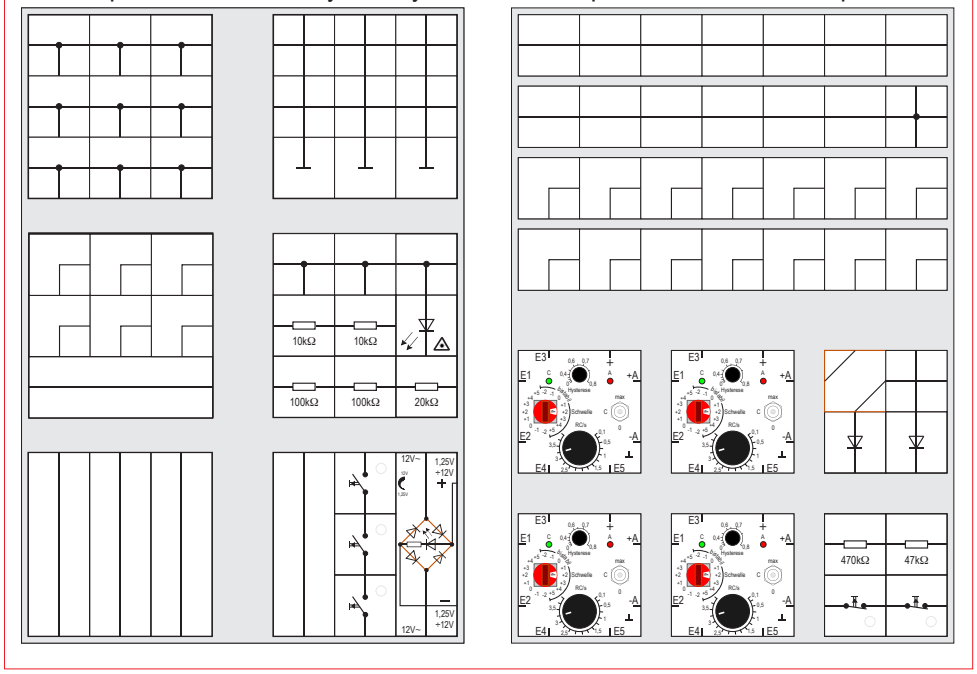
Published 2016 by
Lectron GmbH
Buchrainstr. 18
D-60599 Frankfurt
Tel.: +49 (0)69 90 50 12 82
Fax: +49 (0)69 90 50 12 83
lectron@frankfurter-verein.de
www.lectron.de

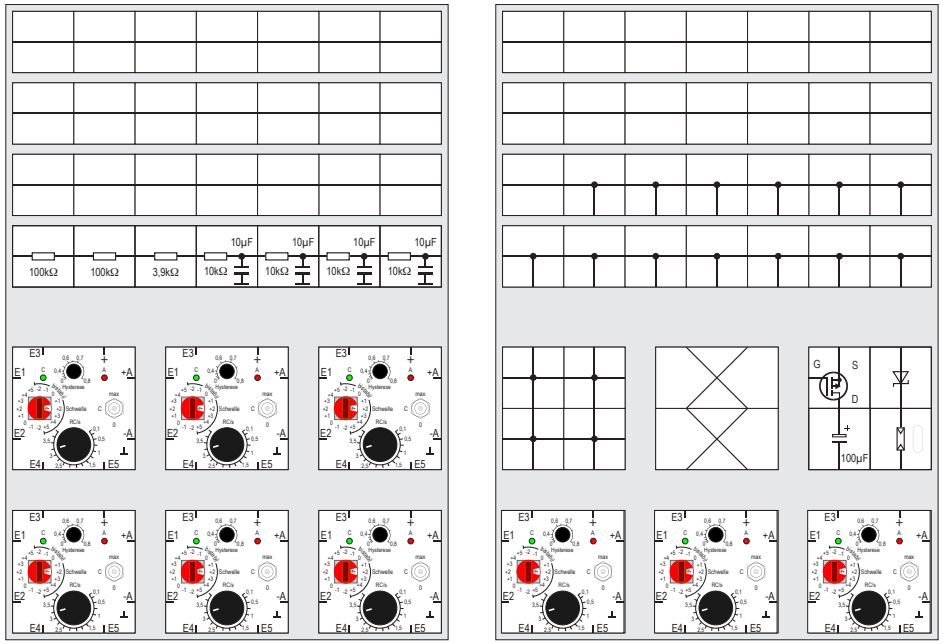
ISBN 978-3-00-051441-8

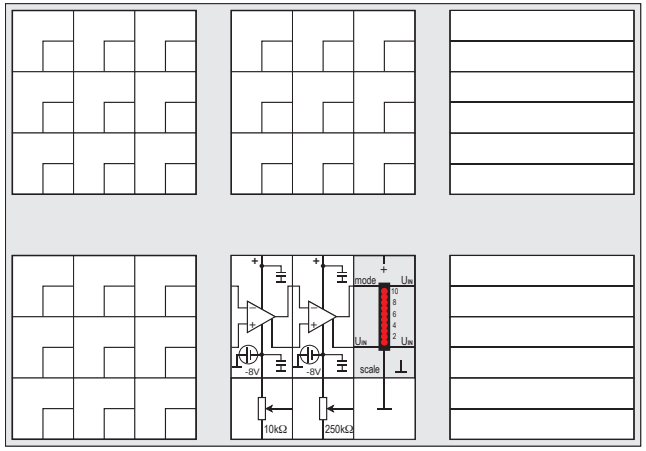
Components list

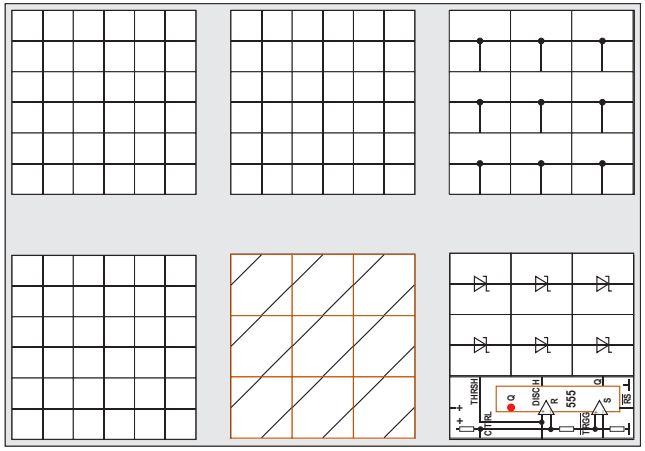
Lectron

For experiments 1 to 23, you only need the components of these two palettes









List of experiments

Lectron

1. Introduction

What is life?

This is a question the physicist Erwin Schrödinger asked in 1944 in his book of the same title, subtitled: »The Physical Aspect of the Living Cell« [1], which sparked interest in the control processes of living cells. By then, physicists were known as experts in »dead matter« and in the properties of atoms and molecules, metals or building materials. But living matter? Skin and muscles? Those are quite unlike stone. Who hasn't fallen from a bicycle and ended up with a skinned knee? What happens then is both wonderful and mysterious: The blood dries and forms a protective layer. Some days later, the layer falls off and the injury below is repaired, brand-new skin shining in its place. What skin cells accomplish here is ordinary to us, but if it weren't, it could be from a clever science fiction novel. The way living cells do this, how their inner control circuits are built and why these lead to meaningful behavior are all part of a fascinating adventure. This is exactly what you can investigate with this LECTRON experimental kit.

Since Schrödinger's book, further advances have been made. Retrospectively, it marks the birth of modern molecular biology. Molecular biologists have figured out many details about the machinery of the living cell, the chemical gears characterizing cell control. Today we not only know that the blueprint of a creature is located inside the nucleus of each of its cells, embodied by a DNA strand, but also know most of the DNA sequence of many organisms, including humans. However, beyond the chemical gears, it can be easy to not see the forest for the trees. Only recently, how this genetic code is executed, how it controls the development of a human, a plant or an animal, started being decrypted piece by piece.

Even how a single living cell controls itself is still mostly unknown and probably the most exciting adventure in today's research. The new field of research called systems biology, which is virtually the think tank of molecular biology, addresses these questions and aims to understand the still incurable diseases being caused by faults in the control system inside the cell.

An important part of cell control consists of genes, a kind of chemical gear, working as switches and forming giant networks of switches toggling each other. These networks ensure, for example, that during reproduction, the cell produces the required proteins, the blueprints for which are stored in its DNA, at the right time, in the right concentration and in the right order.

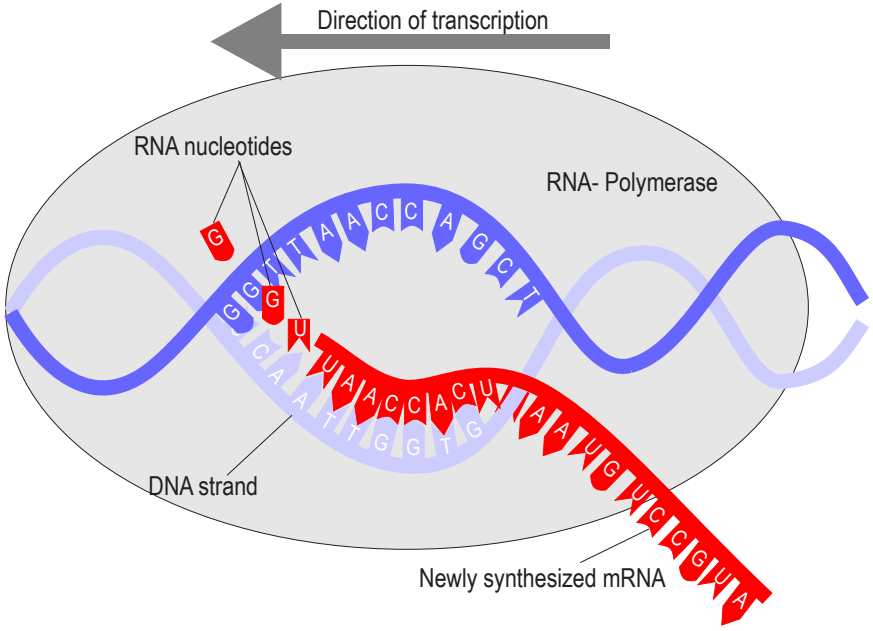
These gene networks form the starting point for the »Gene Regulation« experimental kit. Of course, we do not want to build in detail all of the chemical gears and molecules flitting around inside the cell; that would be far too complicated. With the key element of the experimental kit, the new gene module, we take a more relaxed view and do not worry about the chemical details. Instead, we are very strict about the important switching and regulating properties of genes. As an electronic replica of a gene switch, the module also has buttons and switches, allowing us to set its properties as required, and two lamps telling us its inner state. Together with the lamps of other gene modules, this results in a blinking light pattern, by which we can observe how both a gene network and its genes operate. This is much to the delight of the experimenter, because the spectacle of light has a certain aesthetic charm. In addition to 13 gene modules, the kit contains a power supply and lots of connection modules

for setting up the experiments.
If you already have the LECTRON gene experimental kit, it is time to start! If you do not want to buy the complete, admittedly very big, experimental kit right away, there is another option to start. With the first four gene modules and a smaller number of connection modules, you can build experiments 1 to 23 on a DIN A3 assembly board (the subset is marked in red on the component list). For further experiments, especially for the simulation of large biological networks, you need the additional genes and connection modules and, above all, the bigger (DIN A2) board (see component list). The overview at the end of the handbook tells you the necessary number of modules for each experiment, which can be ordered individually if required.

A short introduction in gene regulation
Transcription and translation

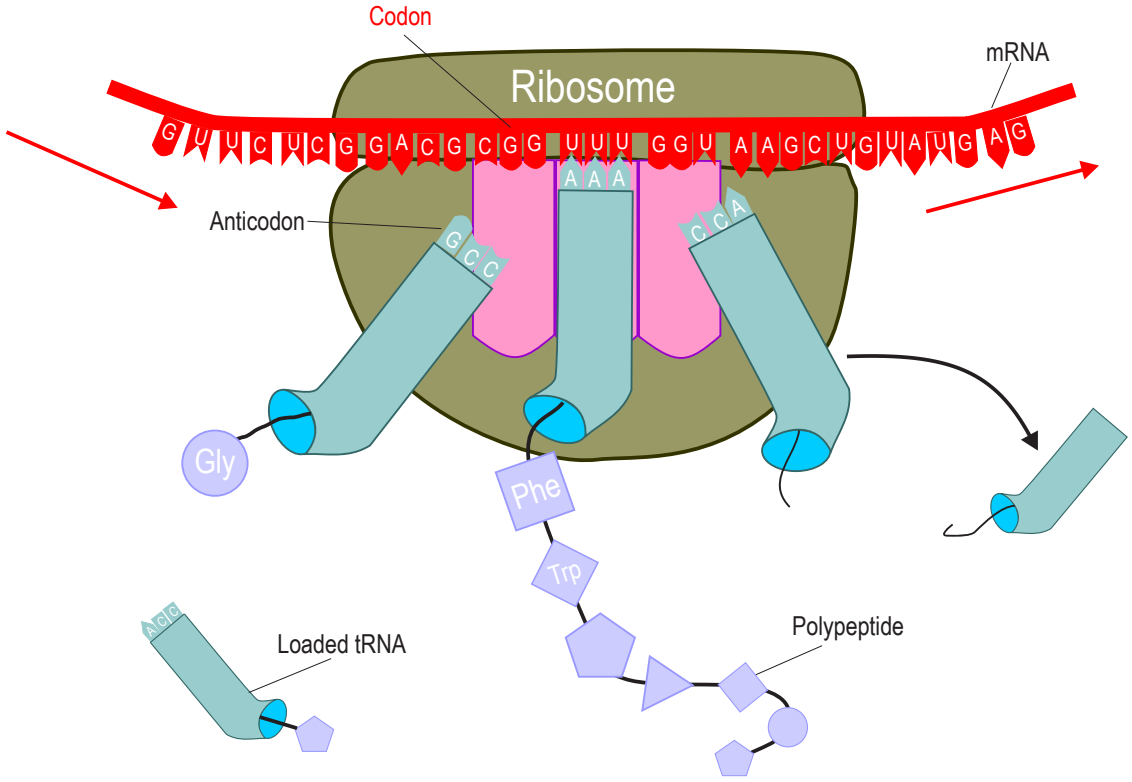
We call our experimental kit »Gene Regulation«, but what do we actually want to model? Let's take a look at the chemical machinery forming the heart of the cell's control. For the following experiments, we fortunately can forget the chemical details right away.
About 60 years ago, the scientists Watson and Crick discovered that the human blueprint is stored chemically on a molecular chain, formed as a double helix structure of two DNA strands, a little like a twisted rope ladder. Both scientists have said that they decided to study genetics because they read Schrödinger's book. By now we know that the information saved on the DNA strand is being read and used by machine-like giant molecules that copy and translate the information, or in the words of a molecular biologist: by »transcription« and »translation« (see figure, page 10). By these processes, certain proteins are derived from the DNA code, and these are the most important construction material of a cell. The DNA segment that codes for a protein is traditionally called a gene. Some of those proteins affect the production of other proteins, for example by activating or inactivating their genes, which is one control process. People often say a gene switches another one on or off, and that is what we should keep in mind for our experiments.
By the way: For more than 90% of the total DNA in a cell, no such translation into proteins was found, so it was considered junk. But current research suggests these apparently useless segments code for RNA molecules, which can complete various tasks.
Within the scope of this handbook, we cannot and will not explicitly describe all the processes initiated by genes. Those who wish to know precise details will find profound material at every level in countless books and on the Internet. A classic is »Molecular Biology of the Cell« [2]. Instead we want to present in a figure how this saved information can lead to formation of a protein. The information flows from DNA through RNA to the protein by the major steps called transcription and translation.
Transcription begins with an enzyme, RNA polymerase, which stretches the DNA double strand to make it readable. During transcription, this enzyme uses the gene for the production of a messenger RNA (mRNA). One of the DNA strands of the double helix serves as a template for the synthesis of an mRNA molecule with a complementary base sequence. Triplets composed of the four nucleotide molecules adenine, abbreviated A, thymine, T (which in RNA is replaced by uracil, U), guanine, G, and cytosine, C, are the code words with which all amino acids can be coded. A always corresponds with T (U in the RNA) and G with C. As an example, the sequence of bases AGT codes for the amino acid serine. In the mRNA, the triplet will be UCA.
In the following step of translation, the genetic information coded in the mRNA is translated into the specific sequence of amino acids that make up a polypeptide, or protein. This happens with another

Lectron



Transcription

Translation



chemical machinery of the cell, the so-called ribosomes. Every codon of the mRNA results in incorporation of one of 20 amino acids into the right position in the polypeptide chain. Start and stop codons mark the beginning and the end of the genetic message. The start in the mRNA is always AUG; the stop codon is always UAA, UAG or UGA.

The amino acids that are required for the setup are brought by transfer RNA (tRNA) molecules, which have an anticodon at one end and a particular amino acid at the other end.

However, the production of proteins is not the whole story. Some proteins can dock to specific segments of DNA, called promoter regions, of specific genes, and thus affect (activate or inhibit) tDNA transcription by the polymerase.

Let's take a look at an example of gene regulation. The answer to the question of how switching genes on and off happens was found by the French scientists Jacob and Monod in 1961. They found out how *Escherichia coli* bacteria accommodate two kinds of sugar as nutrient sources. More precisely, they discovered that some genes form a »genetic switch« together, that switches its digestion between the different kinds of sugar. If it has a choice, *E. coli* prefers glucose for its energy source. For the digestion of lactose, the bacterium needs additional enzymes, whose blueprints are stored in the genes A, B and C (figure page12). Depending on which sugar, lactose, glucose or both, is present, transcription will be activated. To the left of the genes, there is a promoter as a start signal for the copying machine (RNA polymerase) and a fragment of DNA, called an operator, working as a switch. If a perfectly fitting protein binds here, it disables the copying machine.

The following situations can occur:

a) If there is no lactose in the nutrient medium, the enzymes involved in lactose metabolism are suppressed by a repressor protein. The gene on the left is responsible for producing these. The repressor protein binds to the operator region and prevents the transcription of the enzyme genes A, B and C.

b) If there is lactose in the nutrient medium, it binds to the repressor protein. The repressor protein bends after binding lactose and does not fit the operator region any more. This initiates the transcription of the required genes for lactose metabolism, and the cell can then subsist on lactose.

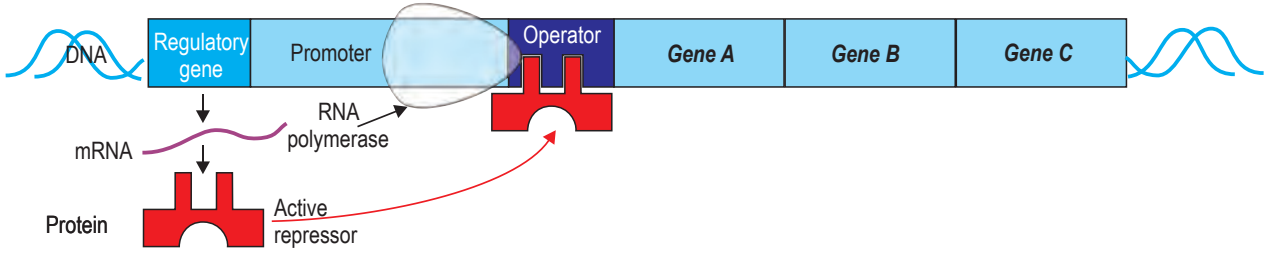
c) If there is no glucose for a while, the bacterium adapts itself longer-term to lactose by producing the »hunger signaling molecule« cAMP. cAMP forms a chemical compound with the gene activator protein CAP, which binds to the promoter and enables the permanent expression of the genes for these enzymes.

d) If both glucose and lactose are in the nutrient medium, cAMP is not produced, and *E. coli* feeds on its favorite food source, glucose, using only a little of the lactose.

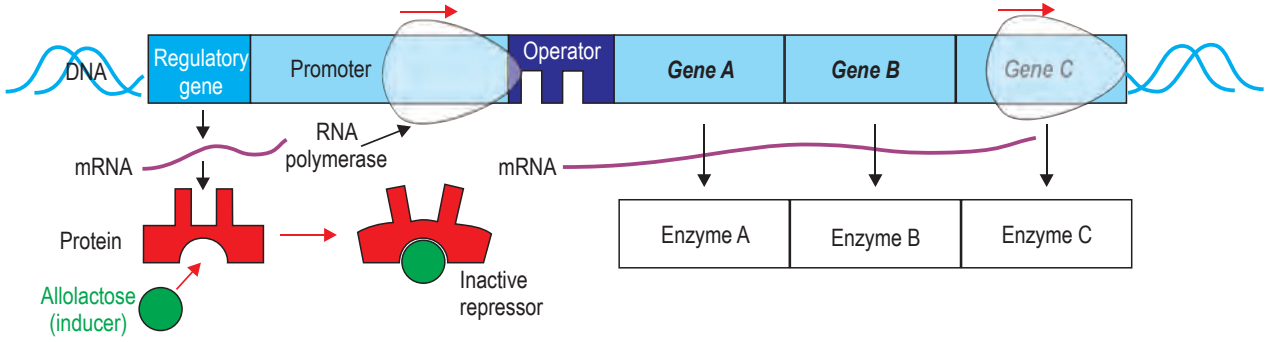
So together we see two switching proteins working: The repressor as an off switch and the CAP protein as an on switch of the enzyme genes.

The transcription of gene A is therefore affected by the activity of other genes, which themselves co- determine the events as so-called transcription factors of the proteins they produce. Since the genes affect each other, a net connecting the genes into a gene regulatory network arises. Transcription factors can activate or inhibit the transcription of genes; these are simple ways they can influence them. However, there are more complex ways containing combinations of transcription factors. For instance, gene A will be activated if factor 1 is available and factor 2 is missing.

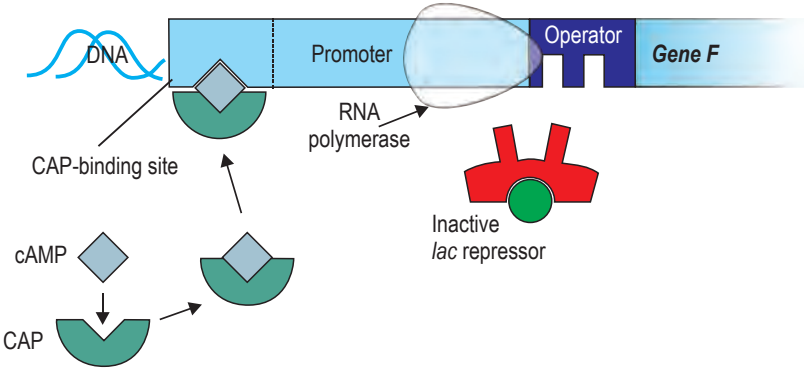
Through experiments, it is possible to find out the interaction of genes and proteins and arrange connection diagrams, just like circuit diagrams. Still, it is extremely difficult to tell if a protein is a transcription factor, and if it is, for what genes and how it controls their transcription. Every systems biolo-



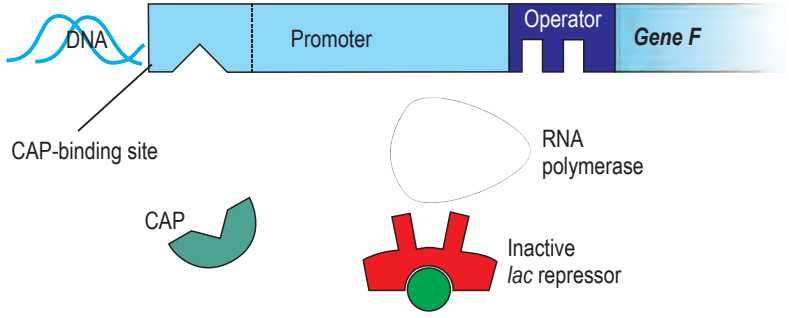
a) Lactose absent, repressor active, genes turned off



b) Lactose present, repressor inactive, genes turned on



c) Lactose present, glucose absent (high cAMP level = »hunger signal«): strong transcription of the genes



d) Lactose present, glucose absent (low cAMP-level): weak transcription of the genes

gist's dream is to hold the whole circuit diagram of a cell in hand, but we are still far away from that. There is too big of a lack of knowledge about the biochemical details of the cell. However, and we will profit from that in our experiments, some biological gene networks are completely known; not for the whole cell, but for some very well-known subprocesses of a cell like the control of proliferation, which has a central role in bread making and beer production. We will simulate such a network with our gene modules.

Models and simulations

What is the connection between control processes in a living cell and electronics? At first sight, not much. If you want to simulate biochemical regulation, you can do so with electronic modules. This is an easy way to build circuit diagrams of genetic networks, and you learn something about the dynamics immediately, without an elaborate computer simulation.

You may wonder whether the outcome could be useful without using serious mathematics like real scientists and instead using toy building sets? We will see in the experiments described below that it is. Two attributes of biological regulation networks will help us: First, the cell has major problems with its »hardware«, or should we say, »wetware«: Imagine building a clock out of a slippery soup of molecules. But this is exactly what the cell accomplishes. Recent studies detected how the cell is able to do that: It links the connections in a way that even inaccurate hardware can handle it [3,4]. That means we can be inaccurate with our models. It is enough if our genetic modules are »approximately« like real genes, and they certainly are. The character of chemical reactions and the chemical switches of cells also help us: The concentration of a certain molecule usually has to exceed a certain threshold before something happens. Such behavior is well-known in engineering and easy to rebuild with electronic components. The outcome is our new gene module: An all-around threshold switch with multiple inputs.

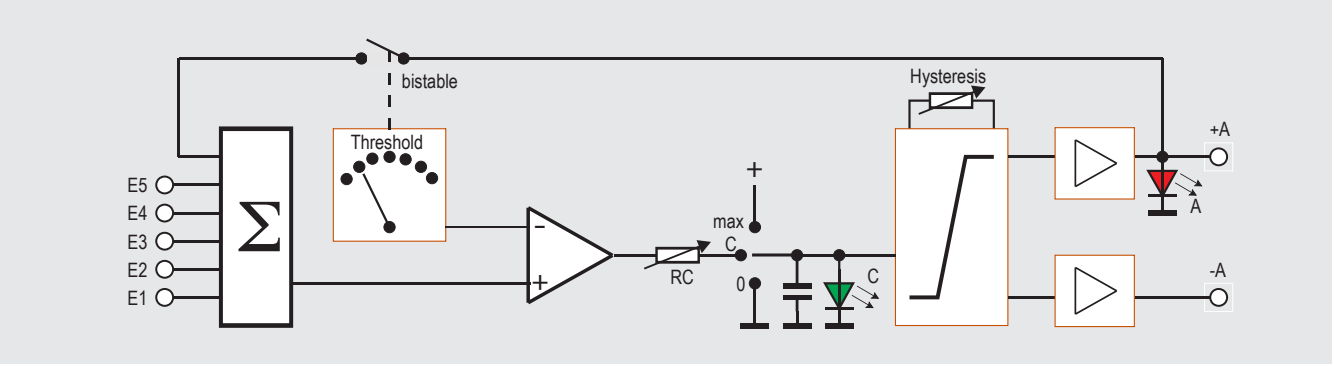
If you are interested in the interior of the gene module, here are some notes. Modules with a variable threshold, which we want to use for modeling genes, are called threshold gates. They work exactly like logical elements in digital technology, with binary input and output signals, so the signals are also described by 0 and 1 in the logical sense.

However, these binary signals are not processed with the NAND or NOR function (so-called Boolean functions) in logical elements. They are simply added: An operational amplifier adds up the input-signal potentials and a comparator compares the sum with the adjusted threshold. If the sum is the same as or less than the threshold, the output potential of the module is 1. If the sum is not large enough, the output is still 0, or it becomes 0 if the potential was 1 before.

Threshold gates or gene modules are very powerful; basically you can attach a specific weighting to the input signals (not necessarily positive integers), and the threshold is also adjustable. To keep the general view, all input signals are the same; they are +1, and the adjustable threshold can therefore just be an integer. On the one hand, these are restrictions, but on the other hand, the simulation of biological networks can be inaccurate, and therefore these are sufficient and make the experiments even easier.

2. The gene module

Block diagram



Before we start the first experiment with the gene module, we want to look inside it. If you do not like technical explanations, just start with the first experiment! The properties of the gene module are intuitive and quickly learned while doing the first experiments. You do not need to know the technical details. Nevertheless, if you want to know them, here they are:

On the block diagram, there are five inputs on the left, which lead to the summing unit Σ. This unit adds up the incoming potentials, considering their signs, which are compared with the threshold potential by the comparator (the circuit symbol is the triangle with + and -). The threshold potential can be adjusted by a turnswitch from -2 to +5.

If the sum of the potential does not reach the threshold potential, nothing happens.

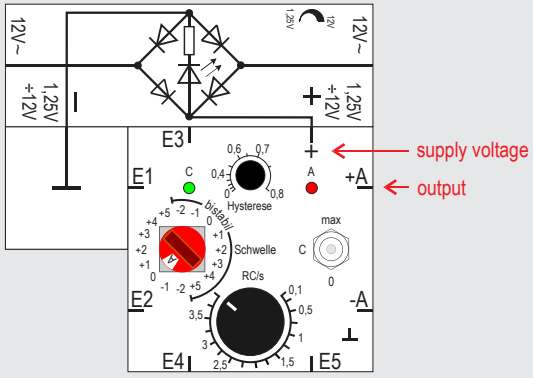
If it is the same or larger, a capacitor with an adjustable resistor begins to charge. The time constant RC is also externally adjustable between 0.25s and 4s.

For all of this a toggle switch with three positions has to be in the center position C.

The brightness of a green LED (light emitting diode) shows the capacitor voltage. The moment the voltage reaches the turn-on threshold of the Schmitt trigger, it will generate a DC signal of +8V at the output +A and another signal of -8V at the output -A. Simultaneously, the red LED will turn on. If the comparator detects at a later point in time that the sum of the potentials is less than the threshold potential, for example because the input signals were altered or the threshold was raised, the capacitor discharges and the green LED simultaneously becomes darker. However, the capacitor voltage needs to further drop below the turn off threshold of the Schmitt trigger now, before +A and -A return to 0V and the red LED turns off. The gap between turn-on and turn-off voltage of the Schmitt trigger, its hysteresis, can also be altered externally between 0% and 80% (0 and 0.8) when connected to a 9V (=100%) supply voltage. 0 means that the turn-on and turn-off thresholds are identical and both are at 4.5V. If the setting is 0.8, than the turn-on threshold is at approximately 8V and the turn-off threshold at approximately 1V.

The capacitor can be abruptly charged to 9V (position »max«) by the toggle switch or discharged to 0V (position »0«).

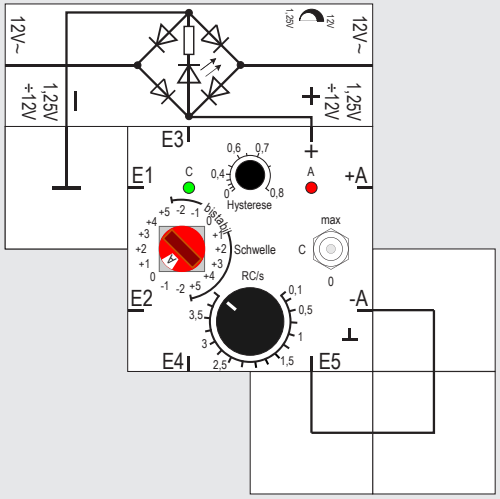
The scale of the turn switch for the comparator threshold is divided in two: In the »bistable« area, the output signal +A is added to the sum of the input signals E1 to E5, but it isn't in the other half of the scale. The reason for that will be given later.



German	English
bistabil	bistable
Hysterese	Hysteresis
Schwelle	Threshold
1,25V	1.25V
0,1	0.1

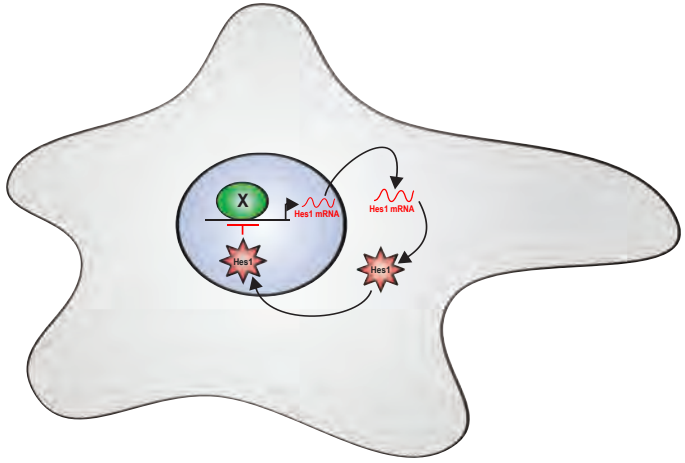
Experiment 1
Activation of the gene module

In the first experiment we want to put the module in operation. Therefore we do not connect anything to the inputs E1 to E5, the switch for the threshold is at position 0 (the outer scale is important, not the signs in the little window), the toggle switch with the three positions is in the central position C, RC = 4s and Hyst = 0.1. The current supply module should be set up in the following way: The positive (+) pole is set on the + symbol of the gene module (note: not on the outputs +A and -A!). The negative pole is connected by the metal plate to the gene module. That is what the so-called ground module is for, which is connected to the pole of the current supply. After connecting the current supply, we see that the green LED C slowly becomes brighter, and after a few seconds, the red LED A begins to glow.
The behavior of the module can be imagined in the following way: All input signals are 0; therefore, the internal sum is 0. The threshold of 0 is reached and the charging of the capacitor begins. Or, expressed electronically: Immediately after the switch on the threshold of the internal Schmitt trigger is reached, the red LED turns on, the +A output gets +8V and the -A output gets -8V. This state is stable and remains this way if nothing is changed.
If we raise the threshold by +1 with the red turning switch, then the internal comparator detects that the sum of input signals, which is still 0, does not add up to the threshold. It switches state and the capacitor starts to discharge, which is shown by the green LED becoming darker.
If the capacitor voltage falls below the turn-off threshold of the Schmitt trigger, the gene module turns both outputs +A and -A off to 0V and the red LED turns off, too.
Turning the threshold to 0 will cause the procedure to be repeated.

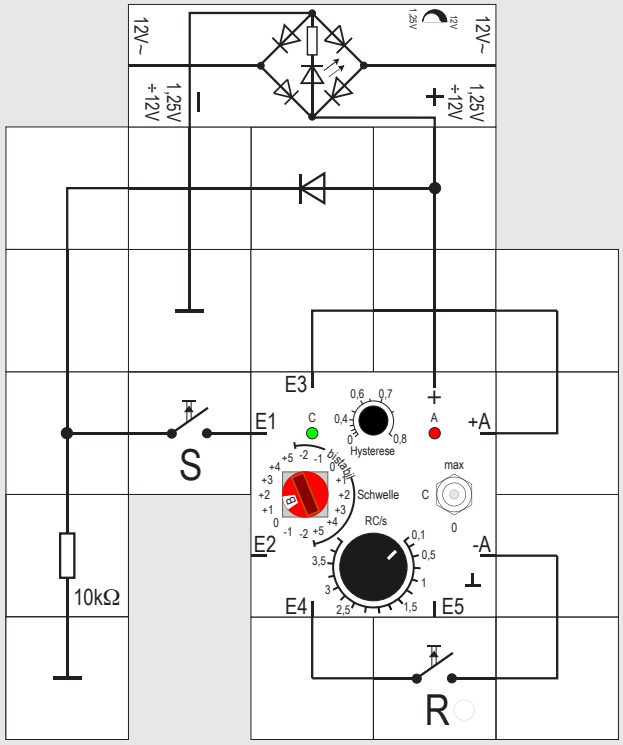

12V~
1,25V
+12V
E3
E1
C
Hysterese
A
+A
Schwelle
max
RC/s
E2
-A
E4
E5

Experiment 2
Oscillator with gene module

We can turn the gene module on and off by constantly changing the threshold between 0 and 1. This procedure can be automated to create an oscillator that works for a period of time without outside help. Here are some general comments:
When the comparator turns on the Schmitt trigger (threshold 0), after a certain period of time it activates the outputs. -A gives -8V, which counts as -1. If we link output -A to any input E1 to E5 of the gene module, the threshold 0 cannot be reached any longer because the sum of all input signals equals -1. As a result, the comparator turns off and the capacitor begins to discharge.
When its voltage gets below the turn-off threshold of the Schmitt trigger, both outputs of the gene module turn to 0, which stops the inhibiting signal, especially of the -A output.
Without this signal, the sum of input signals is 0, and once the input reaches this threshold, the capacitor starts charging again. We have built an oscillator with a frequency that can be chosen by adjustments to the RC circuit and its hysteresis.
The module reflects the fundamental attributes of genes that operate in a living cell as biochemical switches: The slow charging properties and the delayed switching (hysteresis) of the gene module are typical of regulatory properties, among others. The charging of the capacitor imitates the slow synthesis of proteins from the code of the associated gene; transcription and translation take time, sometimes even several minutes. The hysteresis of our modules corresponds to the time between turning on and off. This simulates the delay between genetic switches and signal proteins that is always present.
Genes that oscillate, like our upper blinking circuit, are actually part of nature. Our feeling for day and night rhythm is an impressive example of this effect of genetic control. In fact, our biological clock consists of a little genetic regulatory circuit that is active in certain body cells and gets our body in the right mood for the wake and sleep phases by hormonal signals. The basis of this regulation circuit is a negative feedback loop that follows the principle of oscillators. The figure demonstrates an example of such a biological gene circuit, in this case, a prominent 2-hour clock, which is active in some somatic cells at certain steps in development.



Gene X contains a blueprint for a protein named Hes1. However, this protein inhibits gene X and therefore its own production. This feedback leads to an oscillation of the protein concentration. The dynamic is shown in our experiment. The oscillating brightness of the green LED simulates the changes of protein concentration actually observed in real experiments with cellular clocks.


12V~
1,25V ÷12V
12V~
1,25V ÷12V
E3
E1
C
A
+A
Hysterese
S
bistabil
Schwelle
max
C
RC/s
E2
-A
10kΩ
E4
E5
R

Experiment 3
Memory cell with gene modules

When genes work as switches inside a cell, that can mean not only turning on another gene but also turning it off. In biochemical control processes, these actions are called activation and inhibition, respectively. To simulate this close to reality, a gene module would need additional inhibiting inputs next to the normal ones, having, for example, the weight -1. But then our gene module would need to be very big to have enough inputs. A better solution is weighting all inputs +1 and giving the module a +A and a -A output. The negative output will be connected with the inputs of all the gene modules the gene should inhibit, just like the connection in experiment 2.

Here is another example to show that. This time, we want to build a memory cell. We adjust the threshold to +1 and send a set signal S via a push button and a diode to an input. The diode is for dropping the voltage of the battery from 9V to the high signal level of 8V, as if it were from a +A output. We choose RC = 0.25s and a small hysteresis (0.1) so we won't need to push the button for a long time. When pushing it, the red LED turns on. Sadly, it turns off immediately when we release the button. The circuit has to be self-sustaining. Therefore we connect +A with an input. With this connection, we can release the button and the LED keeps shining. Pressing the button now causes the light to stay on. To reset the memory, we connect -A via another (R-) button with a second input. When pushing this one when the memory is set, the +1 threshold will not be reached any longer because of the inhibiting signal, and the memory will be reset.

So we can construct both an oscillator (no stable state) and a memory cell with two stable states.

This switching behavior of our gene module might remind you of logic modules in computer technology. Our module is a kind of threshold module and can indeed behave like many logic gates. Since we want to study this more profoundly, here is a preliminary note about the notation of logic functions. To be able to distinguish a classical logical Boolean function, named after its inventor, and a threshold function at a glance, it became common to use a special notation that we want to use, too. Angle brackets <> replace the round ones () in these functions, for which the familiar rules of multiplication and addition apply. The brackets enclose the input signals; the turn-on and turn-off thresholds follow > as an index (they differ by 1). [5] For instance, you can write that the LECTRON module works like this::

$$A = \langle E1+E2+E3+E4+E5 \rangle_{5:4}$$

If the threshold is +5, it performs the Boolean AND function with five inputs, E1 to E5,

$$A = E1 \wedge E2 \wedge E3 \wedge E4 \wedge E5$$

By the way

$$A = \langle E1+E2+E3+E4+E5 \rangle_{1:0}$$

is the corresponding Boolean OR function,

$$A = E1 \vee E2 \vee E3 \vee E4 \vee E5$$

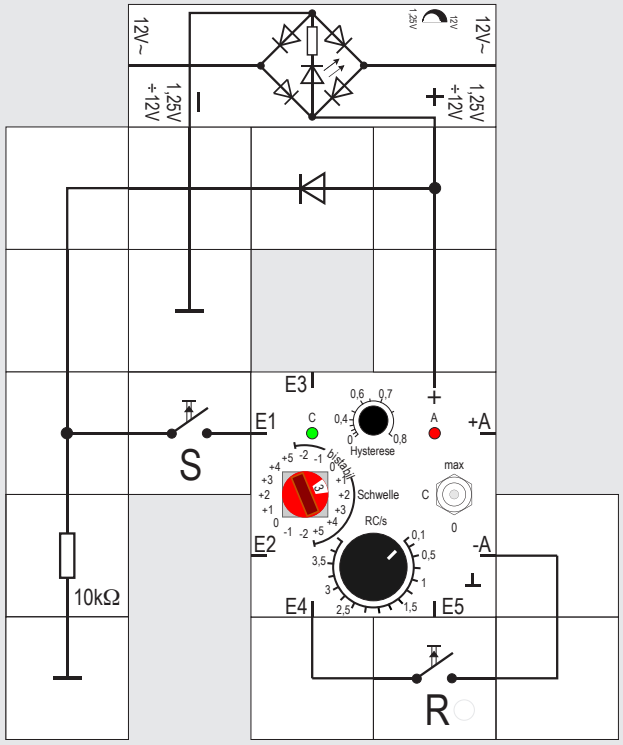
Note: In threshold logic, AND and OR functions only differ in the thresholds; the brackets enclosing the sum of the signals are the same.

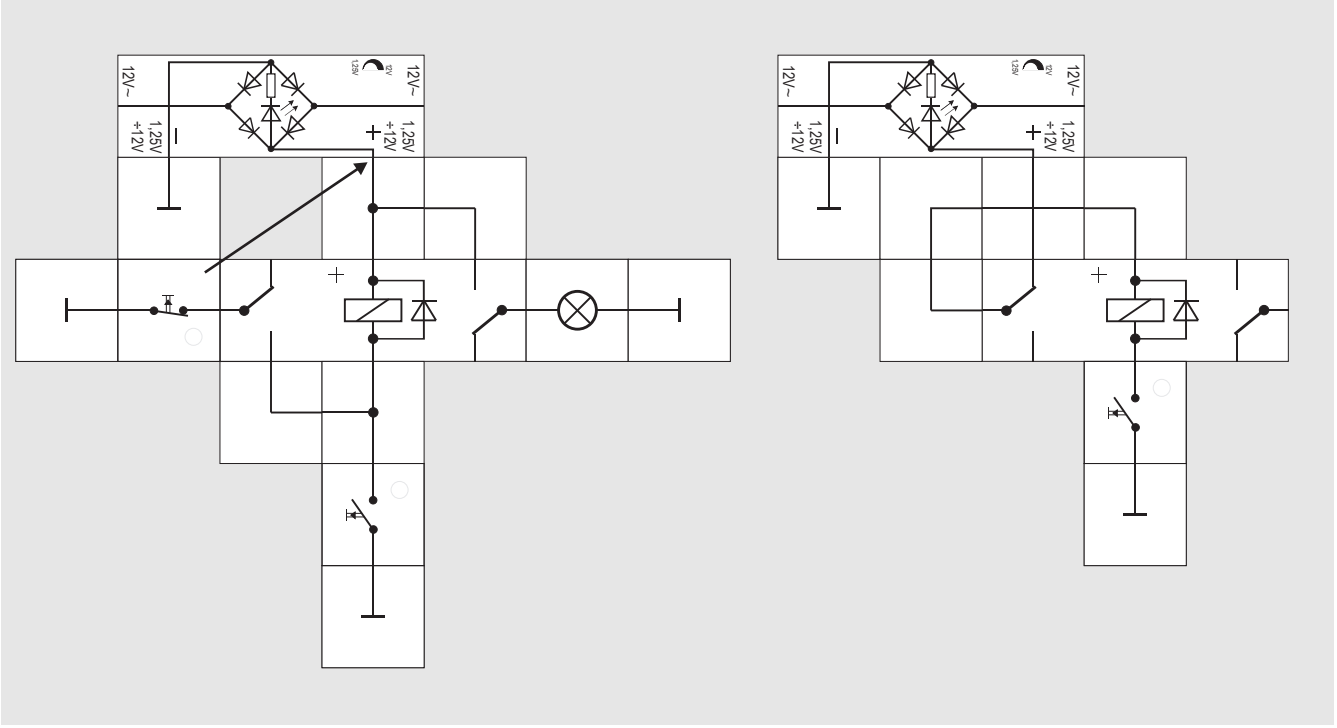
By means of a threshold switch, we can change the function of the module between

$$A = \langle E1+E2+E3+E4+E5 \rangle_{5:4}$$

to $A = \langle E1+E2+E3+E4+E5 \rangle_{-2:-3}$

Negative thresholds exist, too.


12V~
1,25V ÷12V
12V~
1,25V ÷12V
E3
E1
E2
E4
E5
+A
-A
S
R
10kΩ
Hysterese
Schwelle
RC/s
max
bipolar

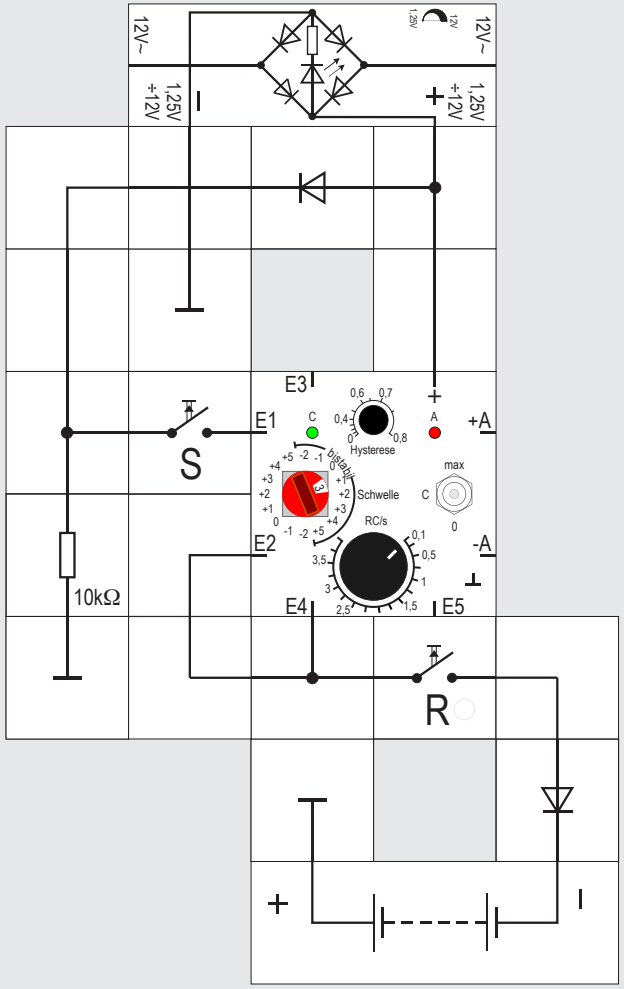


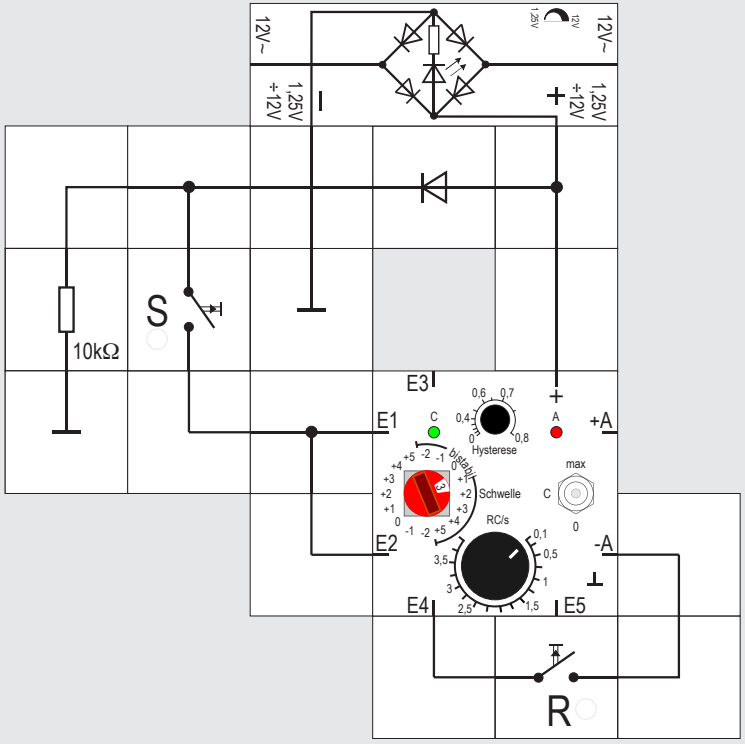
Experiment 4
Memory cell (bistable option)

A memory cell with a gene module is actually easier to construct:
The block diagram on page 15 shows that you can also switch the +A output internally (over a 6th input) with the threshold switch of a gene module to the summing unit if you use the part of the threshold marked as bistable. The external connection is no longer needed, and we save modules.

Mechanical self-sustaining and self interruption analog

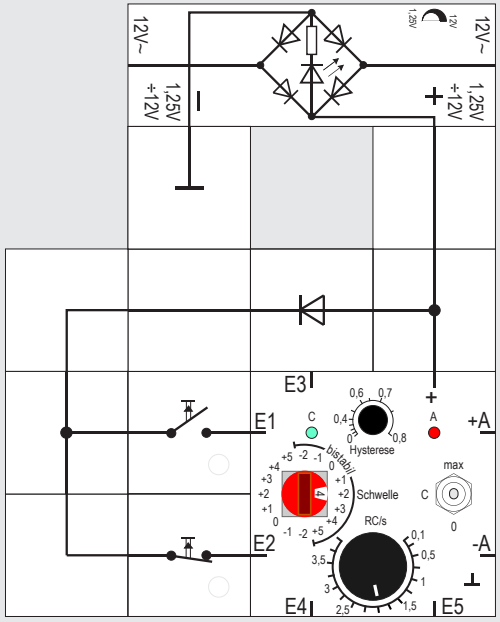
If you have the LECTRON relay module or can get it (order no. 2504), we can construct the mechanical counterpart of this circuit. The lower button sets the relay, which turns the lamp on. The upper button resets the relay by interrupting the lock. If it were arranged just after the positive pole of the battery, it would be dominant (see next experiment).
The right circuit shows an oscillator with a relay (see experiment 2). It is called *Wagner's hammer* and is used for doorbells and buzzers when the relay armature works as a hammer.

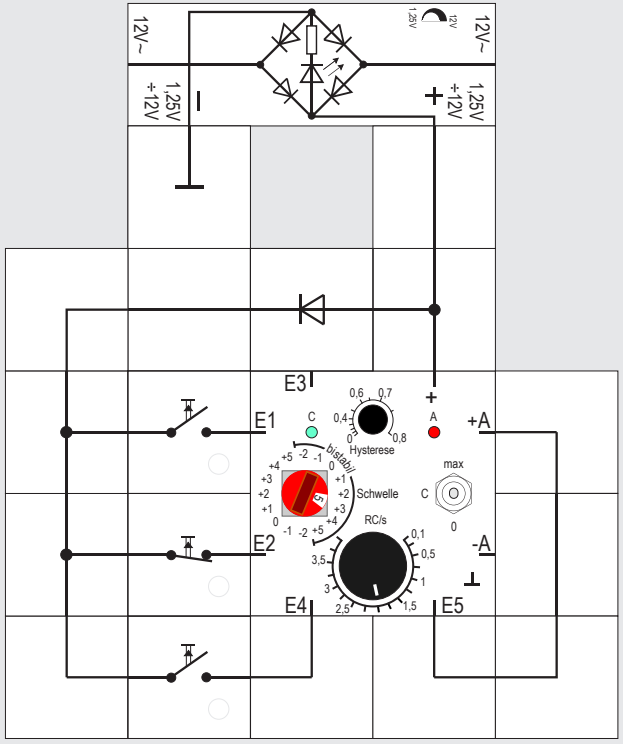

12V~
1,25V
+12V
E1
E2
E3
E4
E5
S
R
10kΩ
+A
-A
Hysterese
Schwelle
RC/s
max


12V~
1,25V
+12V
E1
E2
E3
E4
E5
S
R
10kΩ
+A
-A
Hysterese
Schwelle
RC/s
max

Experiment 5
Dominant set or reset function

The memory cell still has a little drawback: It is not allowed to push both buttons at once because the result depends on which button is randomly released a little later. You may think of this as pedantic, since it is only a small thing, but there will be visible consequences! Indeed, the living cell prepares its own control circuits for possibilities like this by allowing single genes to dominate. These are called canalizing genes.
For our experiment, this means we just have to determine the preceding dominant function, and we are done:
By connecting two inputs with a continuous -8V signal (for example, from a battery or a -A output of a second, permanently turned-on gene module), the R signal prevails when the lower button is pushed. Then the memory cell cannot be reset (left circuit).
The lower diode raises the -9V potential to about -8V.
The opposite configuration is also possible. Just connect the set signal with two inputs. When the upper button is pushed, the memory cannot be reset.


12V~
1,25V
+12V
E1
E2
E3
E4
E5
+A
-A
C
A
Hysterese
bistabil
Schwelle
RC/s
max

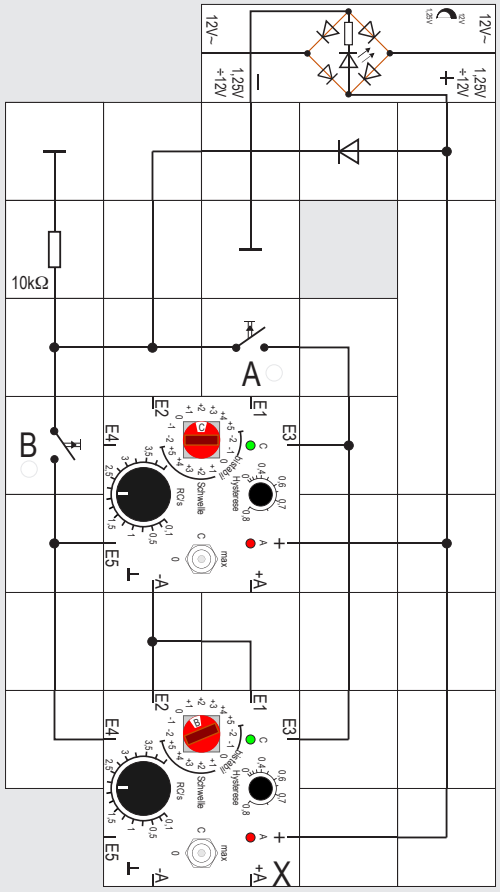

12V~
1,25V
+12V
E1
E2
E3
E4
E5
+A
-A
C
A
Hysterese
bistabil
Schwelle
RC/s
max

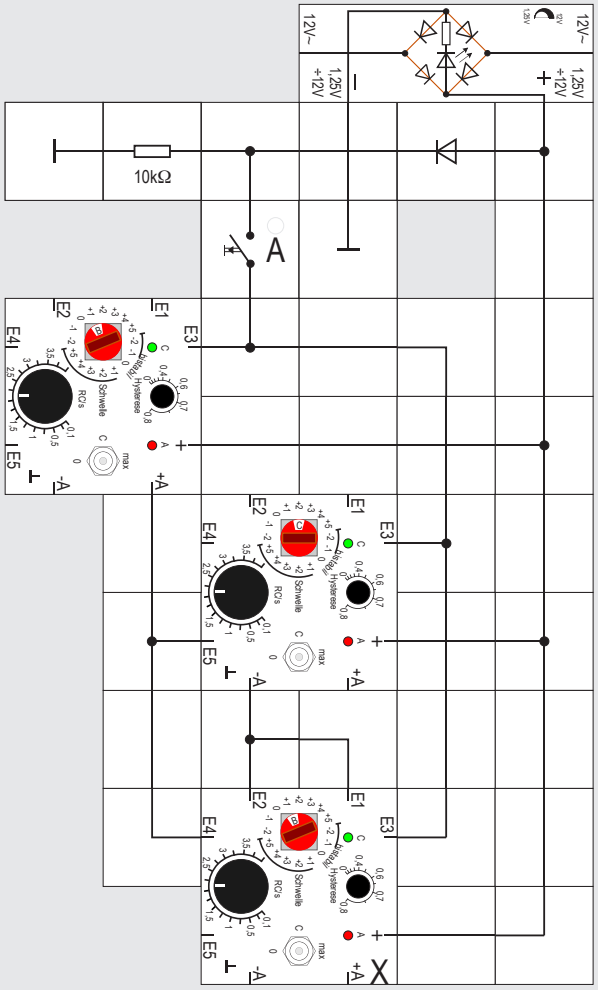
Experiment 5A
Coincidence memory

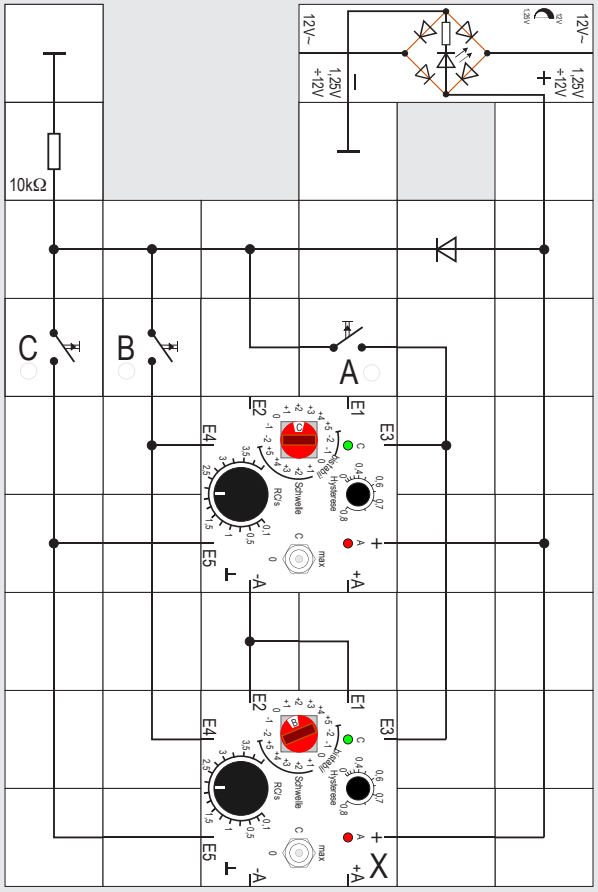
We can also build a memory cell known as a coincidence flip-flop with our gene module. Output +A will be used for summation with a bistable threshold position. We get both input signals of the flip-flop, as we already know, from the supply voltage by dropping it with a diode. They are connected again with the inputs via a push button.
Only if both signals are logical 1 at the same time will the gene module (threshold +2) be activated. It remains in this state because of internal feedback until both signals are logical 0 at the same time. Therefore, we choose a make-contact push button and another break-contact button; suitably, we do not need to push the button all the time to save a logic state. If neither button is pushed, there is no coincidence and the memory cell saves the last state.
According to this principle we can also increase the number of input signals to three; their coincidence will be checked, and if they're the same, they will be saved (right circuit). If we don't push any buttons, there should be no coincidence, so we have to use both kinds of buttons again. Additionally, for the summation, the output has to be double-weighted and the threshold increased to +3 bistable. Therefore, we use both internal feedback and external feedback.

Experiment 6
Realizable threshold functions

The red turnswitch makes our gene module versatile; its threshold function can be used for a number of different logic functions. Let's take another small digression to computer technology to compare our module with Boolean circuits. The first column in the table on page 29 shows the threshold functions that are realizable with the LECTRON module and thresholds larger than zero. You will find the corresponding Boolean expression in the second column. To improve comprehensibility, the input names E1 to E5 of the gene module have deliberately not been used, but instead the letters A to E.
It is usually quite easy to write the Boolean expression with the threshold function in mind. But if necessary, another approach is to write the table of values and the so-called Karnaugh diagram and follow the well-


12V~
1,25V
+12V
10kΩ
A
B
E1
E2
E3
E4
E5
-A
+A
X


12V~
1,25V
+12V
10kΩ
A
E1
E2
E3
E4
E5
-A
+A
X


12V~
1,25V
+12V
10kΩ
C
B
A
E1
E2
E3
E4
E5
-A
+A
X

Threshold function	Boolean function	
$<A+B+C+D+E>_{1:0}$	AvBvCvDvE	OR
$<A+B+C+D+E>_{2:1}$	A(BvCvDvE)vB(CvDvE)v C(DvE)vDE	
$<A+B+C+D+E>_{3:2}$	AB(CvDvE)vAC(DvE)v ADEvBC(DvE)v(BvC)DE = m(A,B,C,D,E)	majority
$<A+B+C+D+E>_{4:3}$	ABC(DvE)vABDEv(AvB)CDE	
$<A+B+C+D+E>_{5:4}$	ABCDE	AND
$<2A+B+C+D>_{2:1}$	A v m(B,C,D)	
$<A+B+C>_{2:1}$	ABvACvBC = m(A,B,C)	
$<2A+B+C>_{2:1}$	AvBC	
$<2A+B+C+D>_{3:2}$	A(BvCvD)vBCD	
$<2A+B+C>_{3:2}$	A(BvC)	
$<A+2B+2C>_{3:2}$	A(BvC)vBC	

known synthesis processes in common textbooks.
The reverse direction requires a lot of experience and intuition. There are synthesis methods, but they are not very easy, so we do not want to present them. In synthesis, you first have to check if the function is realizable with only one threshold module. There are tables about all so-called »linearly separable« functions with a maximum of 7 variables [6,7]. If a function of a variable, for instance, is also dependent on its complement, it is not linearly separable.
As an example we want to show that the in computing very common XOR function is not realizable with only one threshold module. We often encounter this »either-or-function« in everyday life in the form of alternate switching in lamps that can be turned on and off by several switches independently. Their Boolean expression is

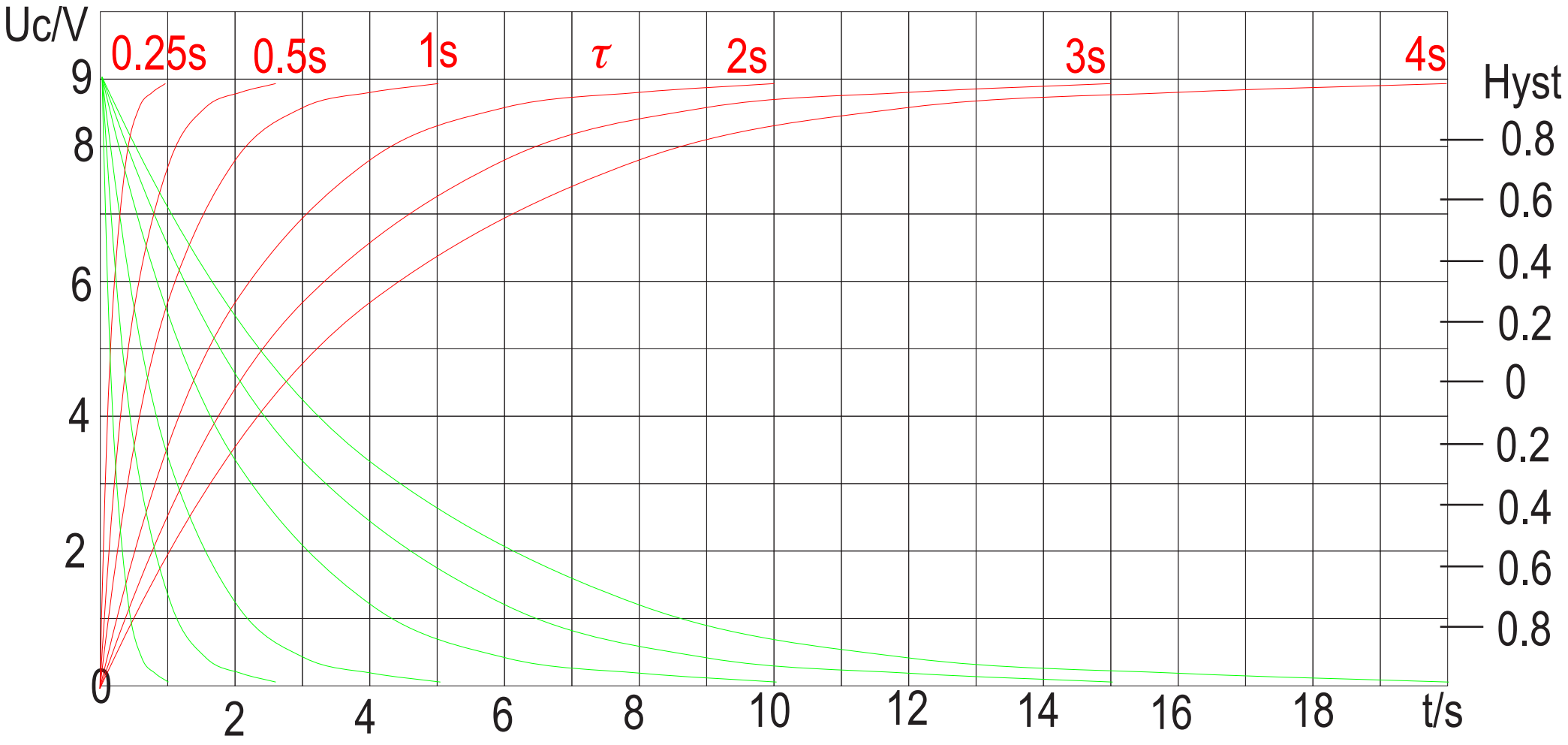
$$X = A\,\overline{B}\ \mathbf{v}\ \overline{A}\,B$$

For its realization we need two NAND modules, a NOR module and two inverters to generate the complements because both variables appear originally and as their complements. So it is quite a bulky function, even in Boolean logic. In threshold logic you need two threshold modules (left circuit). The threshold of the upper one is +2; the lower one is +1. Both pushed buttons each set A and B to logic 1. You will get X at the +A output of the lower module.

On the right, there is a similar circuit for three input signals. X is logic 1, if an odd number of inputs is 1 (odd parity):

$$X = A\,\overline{B}\,\overline{C}\ \mathbf{v}\ \overline{A}\,B\,\overline{C}\ \mathbf{v}\ \overline{A}\,\overline{B}\,C\ \mathbf{v}\ A\,B\,C$$

In the left circuit we can send signal A instead of B to the configuration but now delayed. The delay occurs by an additional gene module, so we get the middle circuit. Now something remarkable happens:
Every time we use only the left button A, no matter if pushed or released, signal X appears for a moment. Its duration of activity is almost equal to the delay adjusted with the additional module. We have to push the button at least until signal X disappears.
This circuit has a biological counterpart that is remarkable. It is often observed in biological processes that just after the activation of a first gene, a second gene begins activity but is immediately repressed by a third gene that has also been activated by the first one. The opposite happens, too. An inhibiting gene releases a second one, which is active for just a short time until being inhibited by a third gene. Our circuit combines both behavior patterns. These circuits are called »feed forward loops«; We' ll refer to them later.


Uc/V
0.25s
0.5s
1s
τ
2s
3s
4s
Hyst
0.8
0.6
0.4
0.2
0
0.2
0.4
0.6
0.8
9
8
6
4
2
0
2
4
6
8
10
12
14
16
18
t/s

Experiment 7
Time constant and hysteresis

In this experiment we want to carefully examine the function of the two rotary knobs on top of the gene module, and therefore we reconstruct experiment 2 (page 18). We already noticed that the internal signal processing needs some time. That is why the delayed output signal only had to be connected to an input as feedback to build a flashing light. With the two rotary knobs, we can adjust the exact properties of the inner signal dynamic.

Using the big button, we adjust the time constant »RC«; that is, the speed of the internal charging, which here adapts the dynamics of the synthesis of a protein encoded by a gene in the cell by charging a capacitor electronically; it is visualized by the green LED. If the sum of the input signals exceeds the adjusted threshold inside the module, the charging of the capacitor begins with the time constant ?τ = RC, adjustable between 0.25s and 4s. The voltage at the capacitor increases over time by the well-known exponential function:

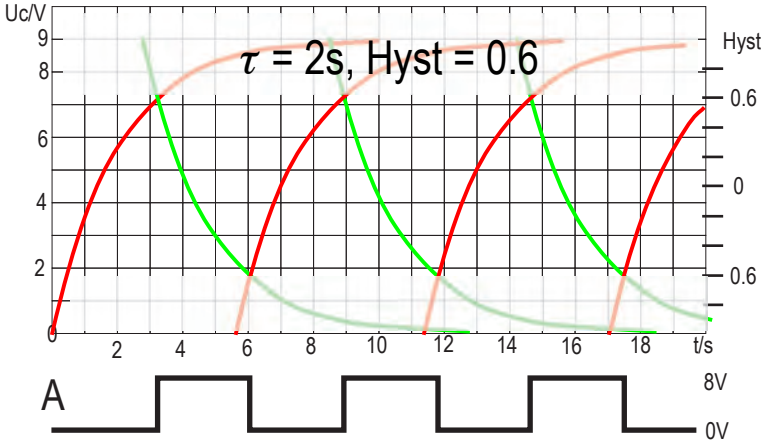
$$U_c(t) = 9V(1 - e^{-\tau/t})$$

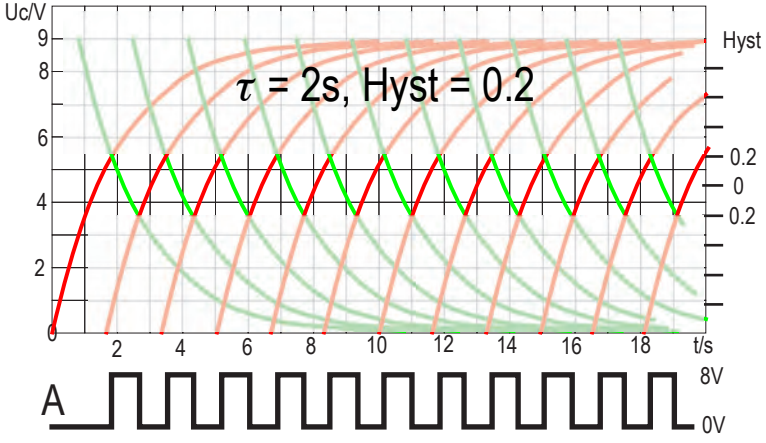
In the diagram above, there are the charging curves for some values of τ. After t = 4τ 98.2% of the final value is always reached, so the capacitor is in fact charged to its maximum.

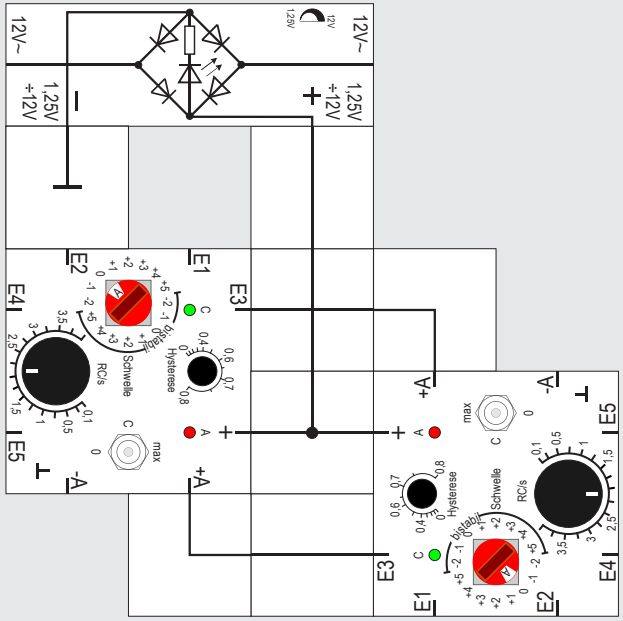
Using the small rotary knob, we can additionally affect »hysteresis«, when a particular concentration of proteins should occur. At a hysteresis of 0, the output of the gene module is activated at a half-charged capacitor. It is deactivated again when the charge falls below 50%. Biochemical compounds often have the property of occupying binding sites longer, even if the concentration of the binding protein has already decreased. We simulate this by hysteresis. At the same time, this has the pleasant side effect of creating something like a time delay of the gene module, and the oscillator does not blink with insane speed, but slowly and clearly. Living cells do have delays like this. In nature, a gene switch often requires a few minutes to achieve full impact.

Now, briefly for the specialists among us, this is how the technology inside the module works: Depending on the adjustable hysteresis between 0 and 0.8, the Schmitt trigger following the capacitor may switch on before the capacitor is fully charged, and output +A becomes logic 1 (8V). If the hysteresis is 0, the turn on and off-thresholds are at the same level of 4.5V.

With a hysteresis greater than 0, the turn-on threshold is at a higher voltage than 4.5V. You can find it in the diagram on the right. The turn-off






12V~
1,25V
÷12V
E1
E2
E3
E4
E5
+A
-A
Schwelle
RC/s
Hysterese
max

threshold moves to smaller values symmetrically. The hysteresis can be calculated by:

$$Hyst = (U_{EIN}-U_{AUS})/9V$$

or $Hyst = R/(25k\Omega+R)$

with $0 \leq R \leq 100k\Omega$

We find that the hysteresis scale at the rotary knob is nonlinear. We can best see the effect of changes of the time constant and the hysteresis at the oscillator circuit in experiment 2 (page 31 right), when $U_c(t)$ runs through the charging (red) and discharging curves (green)

$$U_c(t) = 9V \cdot e^{-\tau/t}$$

periodically. Both curves are displayed in the figure above, too. They illustrate when the capacitor voltage reaches the upper and the lower thresholds of the Schmitt trigger and the influence of the chosen hysteresis (0.2 or 0.6) on the frequency. So the choice of both RC and hysteresis affects the frequency.

3. Small Gene Circuits

Experiment 8
Two coupled gene modules (++)

Now that we have gotten to know the properties and adjustments of the gene module, we want to see what combinations are possible if we connect two of them.

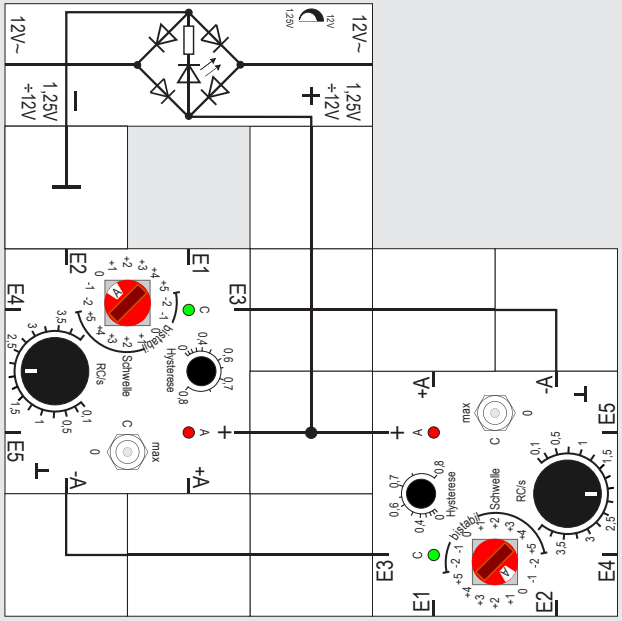
We already used two modules in experiment 6 to construct an XOR; they were, in terms of signal flow, just switched in series, so it was clear what would happen.

In gene regulation, where these modules are used for modeling, this easy configuration does of course happen, too. But there are much more interesting and surprising possibilities if the output signals of the modules influence each other; that is, if there is positive and negative feedback.

We start with connecting the output A of the genes with the input of the other. Since we use activating signals, we call the coupling (++). The RC and hysteresis rotary knobs should be in the middle of their scales.

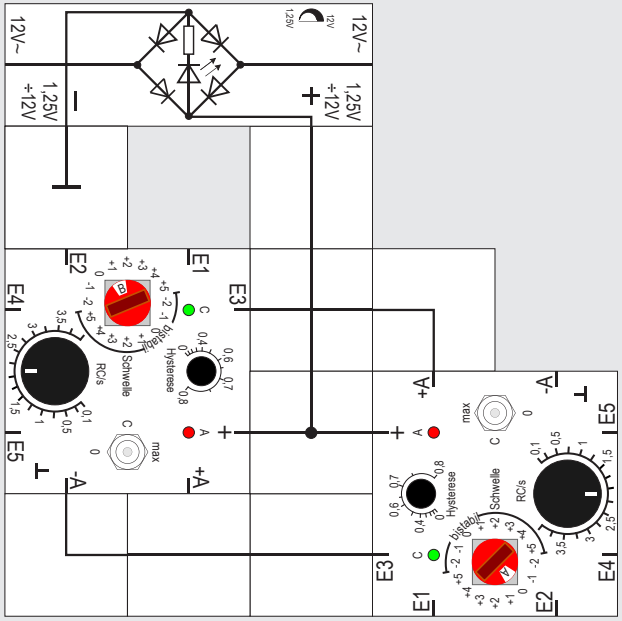
A preliminary consideration on the adjustments of the thresholds: Bigger thresholds than +1 do not make any sense because a single input signal cannot reach it. It cannot fall below -1, either. In the first case, the modules would immediately turn off again if we started them with the toggle switch (the switch knob put for a moment in the »max« position). In the second case we do not need to do anything after applying the power supply. Both thresholds are exceeded and the modules turn on after a short delay and remain in this state. We see the same behavior with both thresholds at 0 or one threshold at 0 and one at +1. The one with the threshold of 0 turns on first and activates the other one.

If we put both thresholds to +1 and put both switches to »max« for a moment, the modules remain active because they activate each other. They remain inactive if both start in the »0« position. In conclusion, there is nothing exciting about this circuit.


12V~
1,25V
÷12V
E1
E2
E3
E4
E5
+A
-A
Schwelle
RC/s
Hysterese
max

Experiment 9
Two coupled gene modules (--)

The behavior of the connection becomes a little more interesting if we couple both threshold modules by their inhibiting outputs (coupling (-)).
Both thresholds should be 0. We move both toggle switches to the »0« position at the same time. Since both thresholds are reached, both capacitors charge and the green LEDs shine brighter and brighter. Even if we try adjusting the hysteresis and time constant RC, we will not be able to turn on both modules exactly at the same time. The one turned on first will immediately turn the other one off by its inhibiting output. The same happens if we start in the »max« position. Both capacitors discharge, shown by the green LEDs becoming darker. But one module will win the race, turn off first and thereby stop the inhibition of the other one. That one will remain on and inhibit its partner permanently.
The result: The (--) coupling gives us a stable state after a short time. One module will remain in the on state and the other one in the off state.

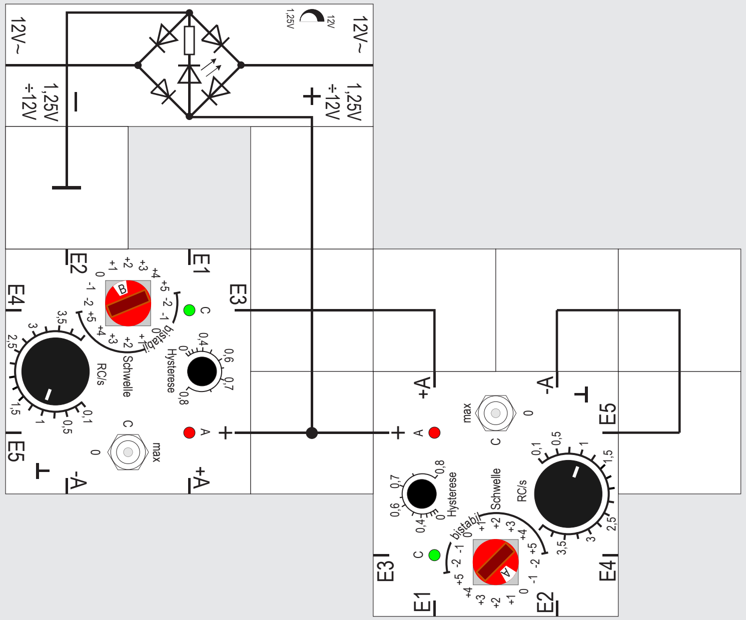

12V~
1,25V
÷12V
E1
E2
E3
E4
E5
+A
-A
Schwelle
RC/s
Hysterese
max

Experiment 10
Two coupled gene modules (+-)

The third possibility is mixed coupling (+-).
We set the thresholds of both modules to +1 and start by putting the toggle switch on only for a brief moment in the »max« position. The modules will turn off one after the other and remain in the stable state, turned off.
In the second part of the experiment we change the thresholds to 0. The start is in the »0« position. We do not have to do anything; the capacitor charges itself because both modules reach the threshold and the inhibiting signal is inactive in the beginning.
No matter what the start position is, after a short time, one module, in our circuit the left one, whose inhibiting -A signal is connected to the other module, will stay turned on and the other one will stay off.
In the third part of the experiment we increase the threshold of the left module to +1. Thus its green LED becomes slowly darker. Its threshold is no longer reached and the capacitor starts to discharge. After a moment the module turns off. So the inhibiting signal to the other module stops and it reaches its threshold. Therefore it turns on after a while and activates the left one again with its +A output. When it turns on, the inhibiting signal will make the capacitor of the right one discharge, and the game starts again. So we built an oscillator, as in experiment 2, but this time a little more complicated; not with one module, but with two modules. By the two red LEDs we can see that the oscillator has four phases: Both turned on, only one on, both off, only the other one on.
Independently of how we start, »max« or »0« or mixed, when both toggle switches are in position »C«, an oscillation will occur. For gene regulation, this means that one gene activates the production of a protein. When the concentration of the protein exceeds a critical level, a second gene will be activated to produce another protein which itself inhibits the production of the first protein; now the concentration of the protein produced by the first gene sinks and the activation of the second gene is stopped, so there is no inhibition of the first gene any longer, and so on.

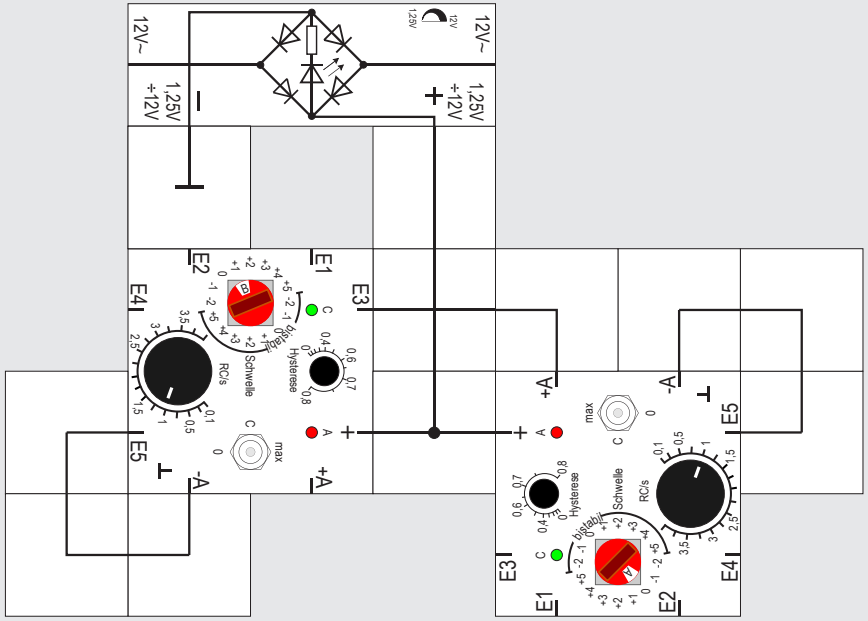
Experiment

11



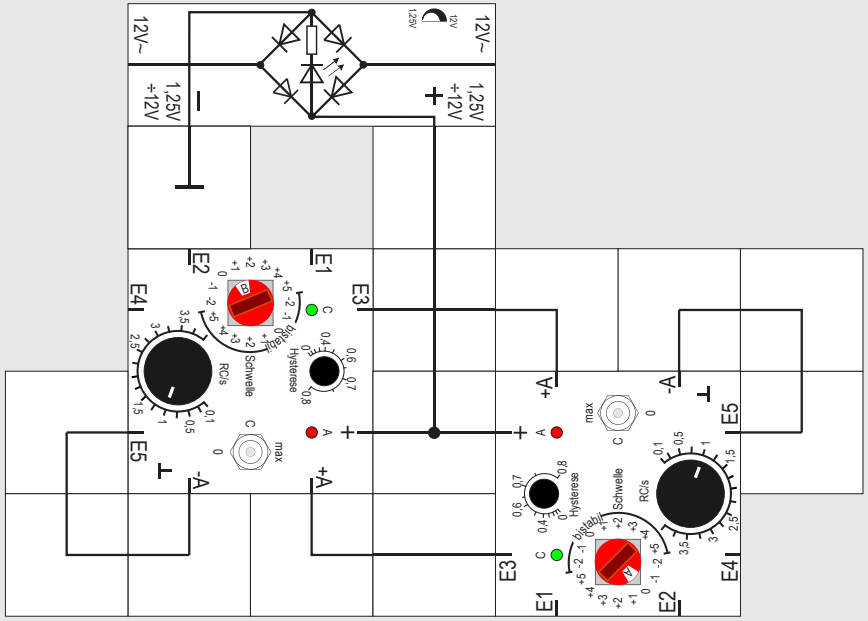
Experiment 11
Two master-slave coupled gene modules

It is possible to couple two modules with the master-slave principle: The right module works as an oscillator with threshold 0, as we know from former experiments. Its oscillation depends on the adjustment of RC and the hysteresis, and it is constant. Both adjustments should be in the middle area. This one is the master. Its +A output leads to the input of the left module, the slave.
The setting of the threshold of the slave is +1; it has to follow the master when the master gives the activation signal. However, this only works if the slave's further adjustments fit the master's. If RC and hysteresis are too high, its state doesn't change, depending on how it was started: If we start from »0«, we can find out from its green LED how the capacitor is charging and discharging, but it is not able to reach the on threshold of its Schmitt trigger, because the red module turned off in the meantime.
If we start from »max«, the capacitor discharges in the time-offs of the right module, but it doesn't reach the off threshold of its Schmitt trigger, because meanwhile the right module has already turned on again. If the left has to follow the right, its adjustments must be adapted.


12V~
1,25V
÷12V
E1
E2
E3
E4
E5
+A
-A
Schwelle
RC/s
Hysterese
max

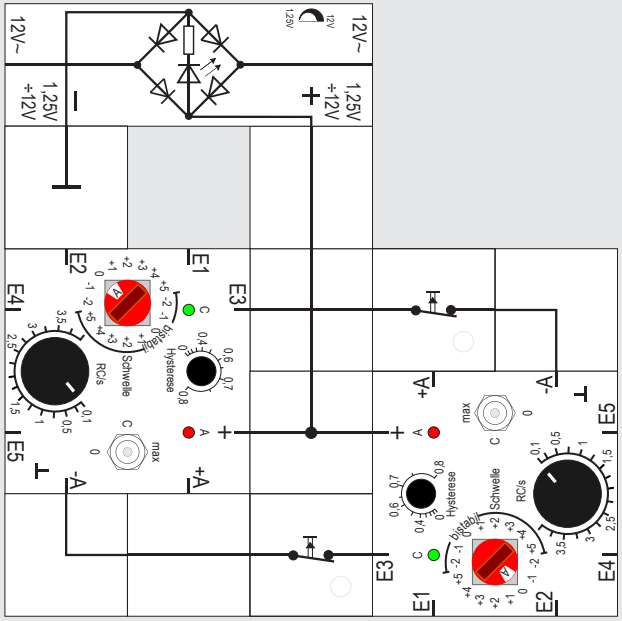
Experiment 12
Two singly coupled oscillators

A very interesting configuration comes into being if we add three angled connection blocks and thus make the left gene module oscillating, too.
After adding the modules we at first cut the connection from the +A output of the right module to the input of the left module and then we adjust its threshold to 0 ,too, so it can work as an oscillator.
We try having both modules oscillating with the same frequency, which doesn't work well. Sooner or later the phase shift will change, caused by a little unavoidable frequency difference.
Therefore we change the adjustments of the left oscillator in such a way that it oscillates just a little quicker. With some intuition and careful observing of the frequencies, it should work.
When we couple the modules again, the right one oscillates with its constant frequency, while we find out that the left module is »disturbed« in its constant oscillation. Sometimes its turn-off times become longer.
If we change the threshold to +1, something remarkable happens: After a short period of time the right module synchronizes the left one. Both are oscillating with the same frequency, but phase shifted.
We should be able to change the phase shift by gently changing the RC adjustment of the left module without losing the synchronization. It is possible to get almost opposite phases with little overlaps of the turn-on times.
There is an adjustment of the oscillation at which the red LEDs turn on at different points of time but turn off simultaneously. When this state is stable, we can lower the threshold of the left module for testing reasons from +1 to 0; then surprisingly the process reverses: The LEDs turn on simultaneously, but turn off at different points of time. The frequency of both oscillations is the same. Only the duty cycle is different.
We change the threshold back to +1. If we also lower the RC values, we can even double the frequency of the left module and it will still oscillate synchronously.
Otherwise, if we raise the RC value, we reach an area where the frequency of the left module is just half of the other.
Thus we are able to synchronize two independently oscillating modules with »similar« frequencies by only one link from the output of one to the input of the other.
However, if the frequency adjustments are too different, the left module doesn't make it turn on again if $f_{left} < f_{right}$; but if $f_{left} > f_{right}$, the right module releases the oscillation for a short period of time and then turns it off.


12V~
1,25V
÷12V
E1
E2
E3
E4
E5
+A
-A
Schwelle
RC/s
Hysterese
max

Experiment 13
Two reciprocally coupled oscillators

Now we want to go one step further, so we connect the output of the left module with the input of the right. The threshold of the left one is +1 and that of the right is 0. Even when the oscillators in the last experiment nicely oscillated at similar frequencies, this experimental setup behaves totally different with this little adjustment: It seems that you can recognize a certain complex pattern (for example, the right LED turns on and the left blinks four times, the right blinks and the same recurs again and again), but suddenly everything changes.
If you wait for another moment, it seems that this complex pattern is repeating but then it changes again. It is insane: Although the green LEDs offer support, it is difficult to predict when the red LED turns on and off. Furthermore, tiny changes (preferably the RC adjustment) alter the pattern.


12V~
1,25V
÷12V
E1
E2
E3
E4
E5
+A
-A
Schwelle
RC/s
Hysterese
max

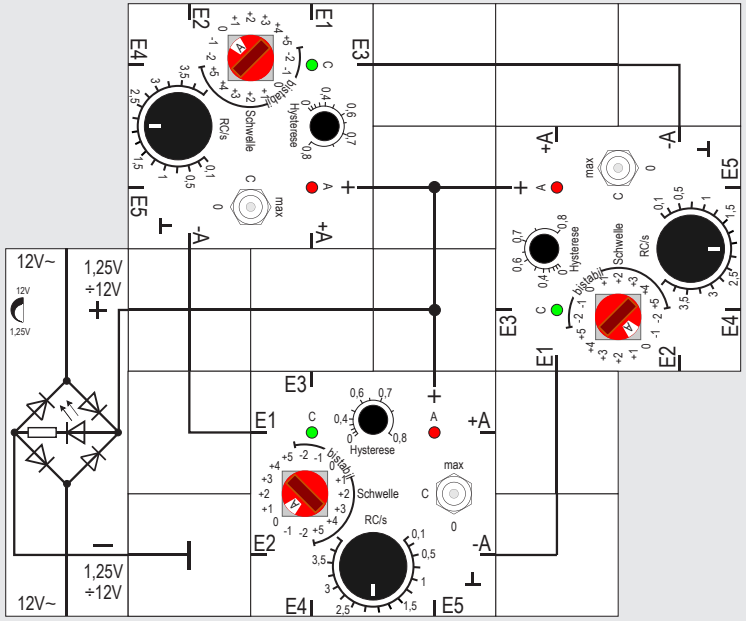
Experiment 14
Toggle switch

At the end of the experimental setups with two gene modules we step back to the (--) coupling of experiment 9. However, this time we add a normally closed push button (nc) to both connections. The RC adjustment at both modules should be 0.1s and the hysteresis 0.1, so they switch fast.

As long as we don't touch the buttons, this experiment is the same as experiment 9: After applying the supply voltage, one module turns on, one remains off. Which one is on and which is off cannot be predicted.

If we now push the button connected with the -A output of the module whose LED is on, that LED turns off and the LED of the other module turns on. The inhibiting signal is deactivated.

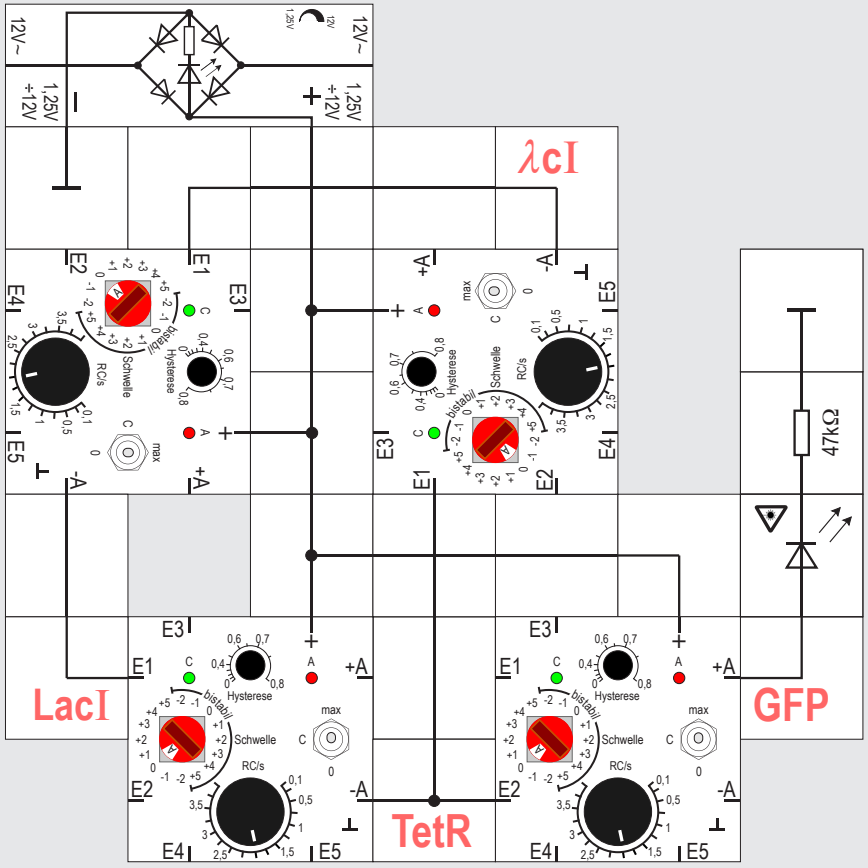
Afterward we can do the same with the other push button, and the converse happens. The starting situation is re-created. With two switches that must not be pushed at the same time, we have built a sort of a toggle switch. That such a toggle switch actually works in a living cell was shown genetically by three US scientists in the year 2000 [8]: Without much difficulty, they took two genes with known protein and binding sides, combined them like the connection pattern above and introduced the whole construction into *E. coli* bacteria. By chemical signaling substances, they were able to turn the toggle switch in the bacterium back and forth and to observe the process by means of a green florescent protein, similar to the flashing green LED in our experiment.


12V~
1,25V
÷12V
+
−
1,25V
÷12V
12V~
E1
E2
E3
E4
E5
+A
-A
Schwelle
RC/s
Hysterese
bistabil
max

Experiment 15
Coupling of three modules (---)

To this point we have examined a lot of combinations of two gene modules, and by doing this we observed surprisingly different circuit dynamics. What will happen now if we add to our two-part combination of gene modules a third? We will see that some new properties will appear that are also interesting from the biological point of view. Before we dash into real-world biological gene networks, we want to examine how configurations of three gene modules act. So we start with a ring connection. There is always a connection from the -A output of one module to an input of the next. All thresholds are 0. Hysteresis and RC adjustments of the three modules are in the middle area.
After turning on the power supply, the capacitor starts charging immediately in the three modules, because all 0 thresholds are reached and an inhibiting signal is not being sent yet.

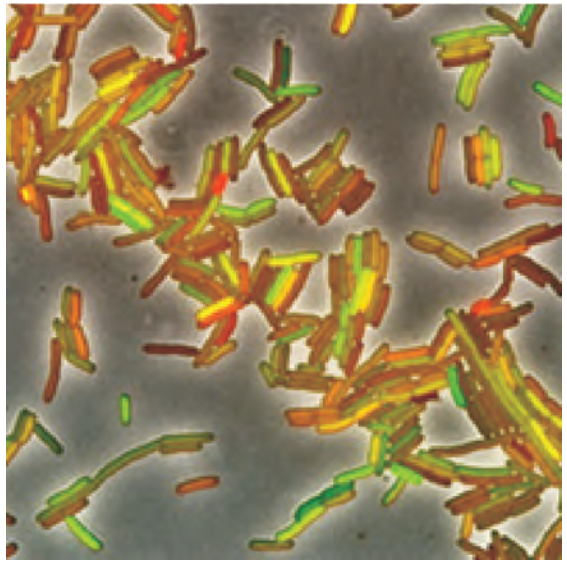
Even if we try, it is not possible to turn on the three modules at exactly the same time; one will be the first. That means it sends the inhibiting signal first and therefore the next in the ring (no. 2) cannot keep charging its capacitor. So this one is not able to send its inhibiting signal, and its capacitor discharges.
As a result the capacitor of the third module will charge completely until its output turns on and sends an inhibiting signal to the first, which causes the discharging of this one. After a short period of time the inhibiting signal of number 1 turns off and the second module can start the charging of its capacitor. The oscillation begins, which is recognizable by a dark LED (sometimes even two at the same time) »circulating« in the opposite direction.
Although at the start of the experiment we thought that nothing would happen because of the mutual inhibition, the opposite occurred.
It is exciting to ask if this type of circuit works in a living cell. That question was also discussed by the two scientists M. Elowitz and S. Leibler, who constructed such a circuit inside *E. coli* [9]. As a matter of fact, using a fluorescent protein, they observed a circulating signal and they called this successful circuit a »repressilator«. We will assemble it in the next experiment.


12V~
1,25V
+12V
λcI
LacI
TetR
GFP
47kΩ
Hysterese
Schwelle
RC/s
bistabil
max
E1
E2
E3
E4
E5
+A
-A

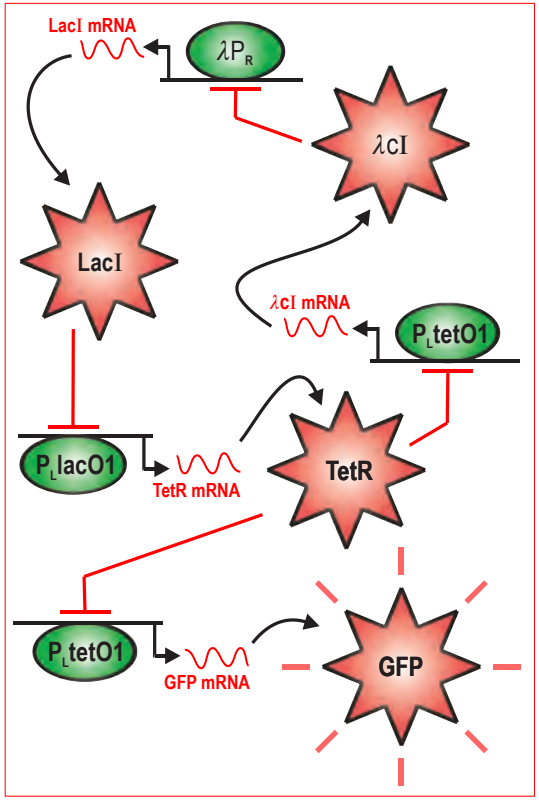
Experiment 16
Repressilator

The repressilator in *E. coli* is made of a genetically built ring oscillator with three genes and at least one additional gene that generates a signaling protein so that it is possible to observe the oscillation. The ring contains three genes, *TetR*, *LacI*, and *λcI*, which inhibit each other, generating an oscillation. This oscillator produces the dye GFP (green fluorescent protein), which leads to a blinking of the bacterium with a 3-hour period. The uncoupling from this »triple circle« happens by another gene module. TetR doesn't just inhibit the production of λcI; it also inhibits the production of GFP by the fourth module.

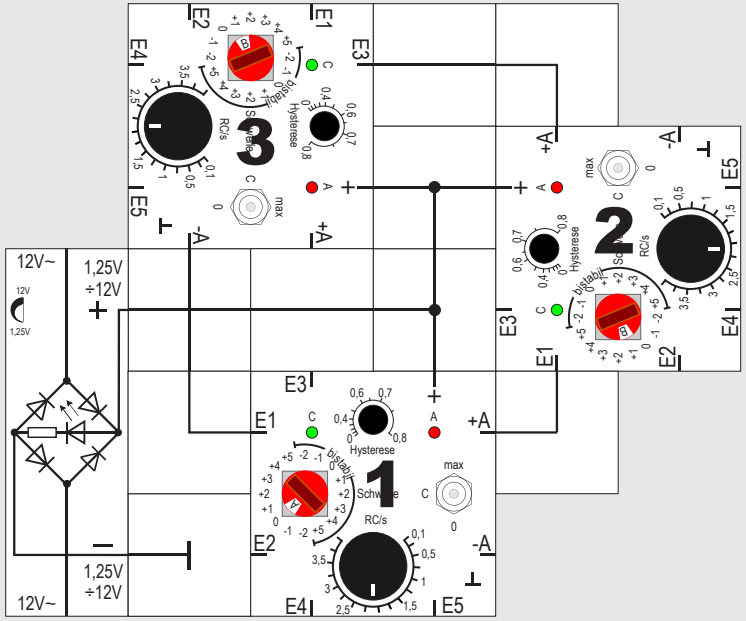
This circuit is quickly assembled with our gene modules; it simulates the observed dynamics of the mutual gene activities. The green LEDs C simulate the constant change in concentration of the single protein by their brightness. The extra LED, which is connected to the fourth gene module, blinks with the same period as the red LED of the module and simulates the GFP. However, the period of oscillation in our experimental setup doesn't take 3 hours. Fortunately for us it is approximately 30 seconds.



E.coli bacteria



Repressilator


12V~
1,25V
÷12V
12V~
1,25V
÷12V
E1
E2
E3
E4
E5
+A
-A
1
2
3
Schwelle
Hysterese
RC/s
max
bistabil

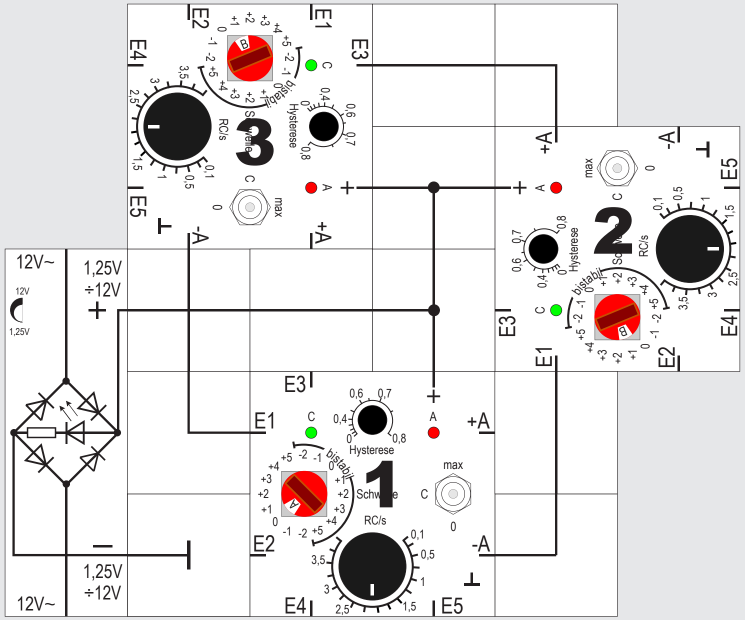
Experiment 17
Coupling of three modules (++-)

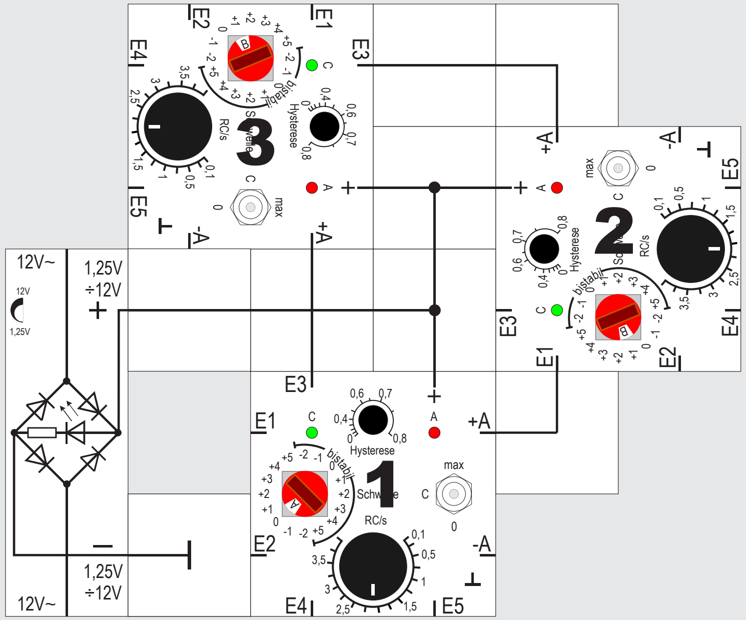
We continue our examination and change the circuit of experiment 15 by sending only one inhibiting signal to the lower module and letting the other two connections start from the activating outputs.

In order to see anything interesting, the threshold of the modules that receive the activating signals must be +1. The threshold of the third (lower) module remains 0.
If we apply the supply voltage, only this module starts to charge its capacitor because it is the only one with a threshold exceeding 0. After a moment, it activates partner no. 2, so this one starts charging its capacitor, too. Its threshold of +1 is reached.
It will soon turn the red LED on and send an activating signal to no. 1, whose threshold will be reached as well, and finally all modules are turned on.
This, however, means module no. 1 sends an inhibiting signal to no. 2, beginning the discharging of the capacitor. As a consequence it turns off. One by one, the activating signals of the others disappear and finally all modules turn off. This is the initial situation and the game starts again.
This configuration shows a circulating pattern as well. It resembles a ring shift register in data processing assembled with three D flip-flops, where the $\overline{Q}$ output of one of the three cells is connected to the D input of the following cell. There is a big difference though. All the cells of the shift register get a central clock. In our experimental setup, however, the time adjustments of the modules determine the time of the circulation.

Experiment

18



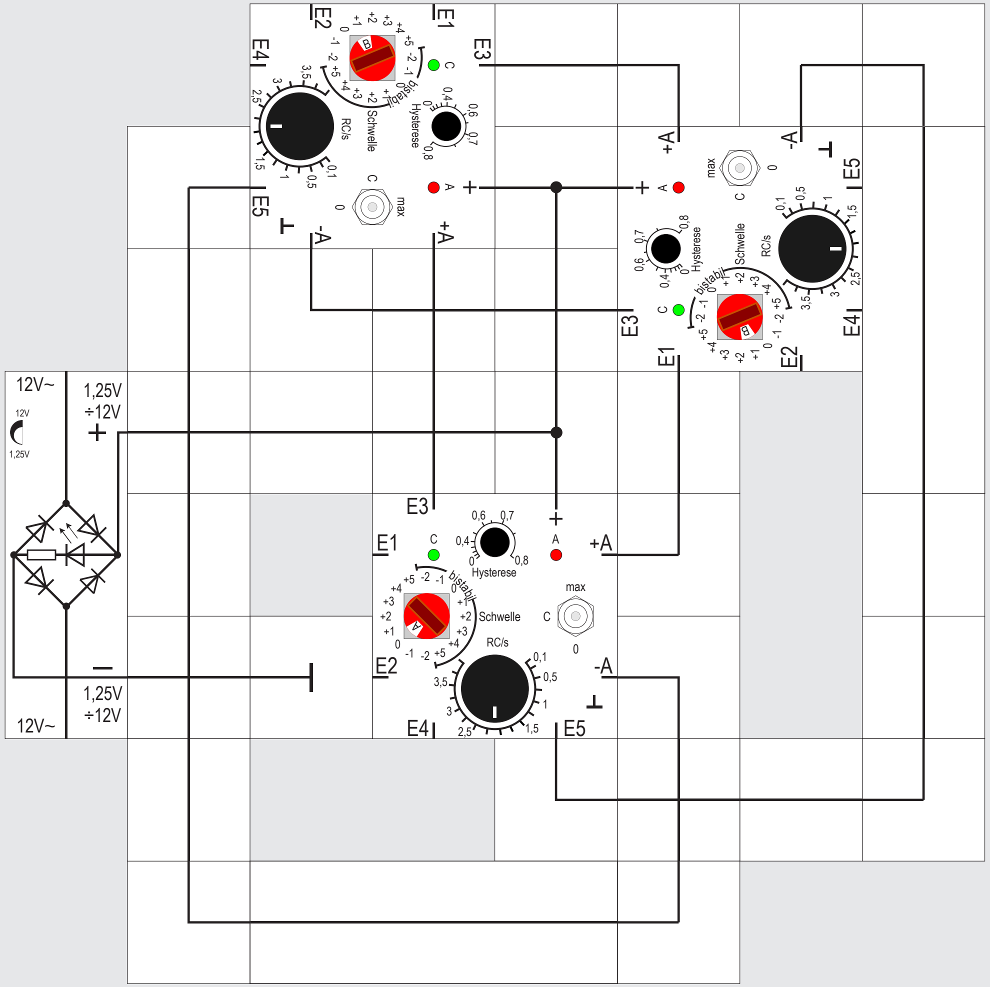


Experiment 18
Coupling of three modules (+--) & (+++)

If we change the active coupling of module one to module two into an inhibiting signal, we observe something surprising: The circulating pattern is gone (see left circuit)! Depending on the adjusted time conditions and on which module we decide to start, we get a stable LED picture after a short period of time that doesn't change: Either module number one is turned on permanently or modules two and three are active.
Now try changing the time adjustments to increase the time at which the state is unstable. Are you able to get a whole circulation of the signal?
Likewise, you can change the threshold to +1 at the (+++) coupling with all three modules to get a stable pattern (see right circuit). If we turn on the power supply, nothing happens at first; all three modules remain turned off because there is no active signal. This state is stable.
There is another state that ensues after a short period of time if you use the toggle switch (briefly turn on »max«) to start the process. At first, you observe a circulating pattern, but after another short period of time, the second stable state appears. All three modules are turned off and remain stable. However, it is worth finding some adjustments to increase the time until this second state becomes stable.
If we compare the different ring circuits set up with the gene modules (---, ++-, +--, +++), we notice something else: We will get a full oscillation only if the number of inhibiting connections is odd. Otherwise the blinking dies out and the state becomes stable. Theoretically this is valid for ring circuits of any size. If you like, try it with some more gene modules!
By the way, this rule applies especially to biological regulatory circuits because every gene has its own pace and quirks, which is very different from computer technology, where logical circuits get their clock from a central timer.

Experiment

19



Experiment 19
Double coupling of three modules (± ± ±)

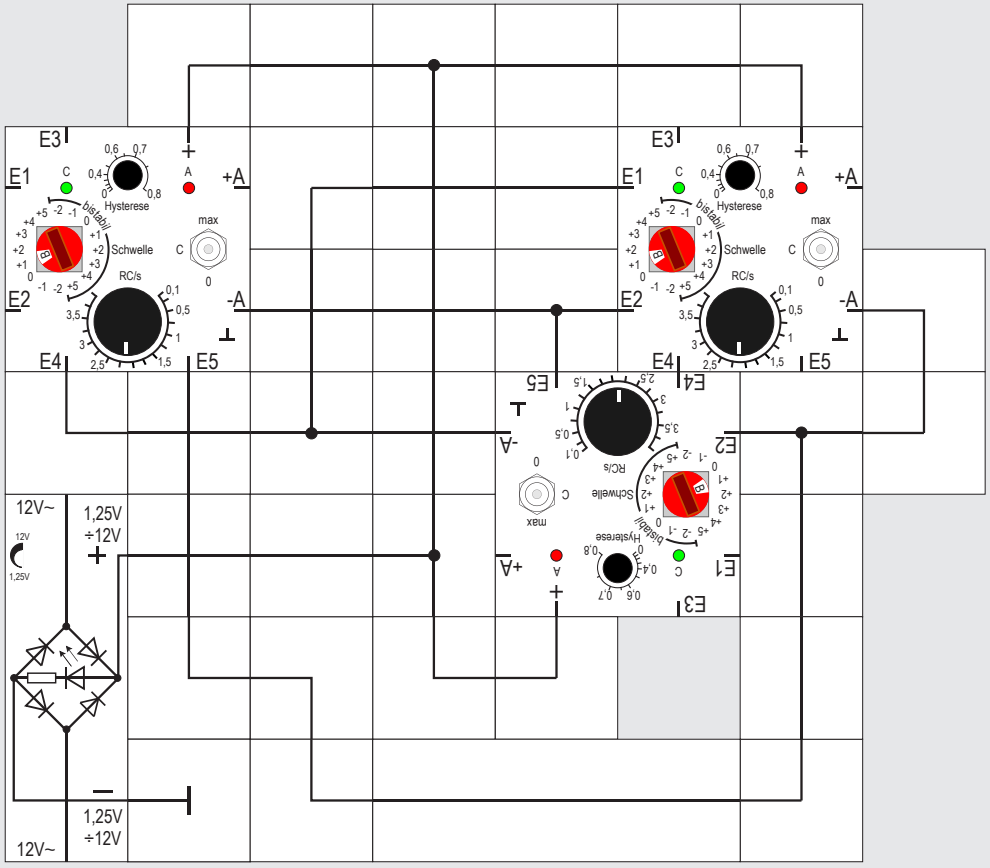
There are more possibilities for connecting three modules than for two. We want to continue the experiments with three modules and a setup of identical couplings. Every module in the ring will activate its successor and inhibit its predecessor. Then we get a sort of double coupling. All three modules are equal.

If all modules have about the same time constant and the threshold +1, nothing will happen when we apply the supply voltage. All of them remain turned off. Even if we start all of them in the »max« position and switch them to »C« at the same time as far as we can, they all turn off after a while.

If we start only one of the modules, at first there is something like a circulating pattern, but it dies after a few steps. We can extend its lifetime by carefully changing the time constants, but we cannot make it last indefinitely.

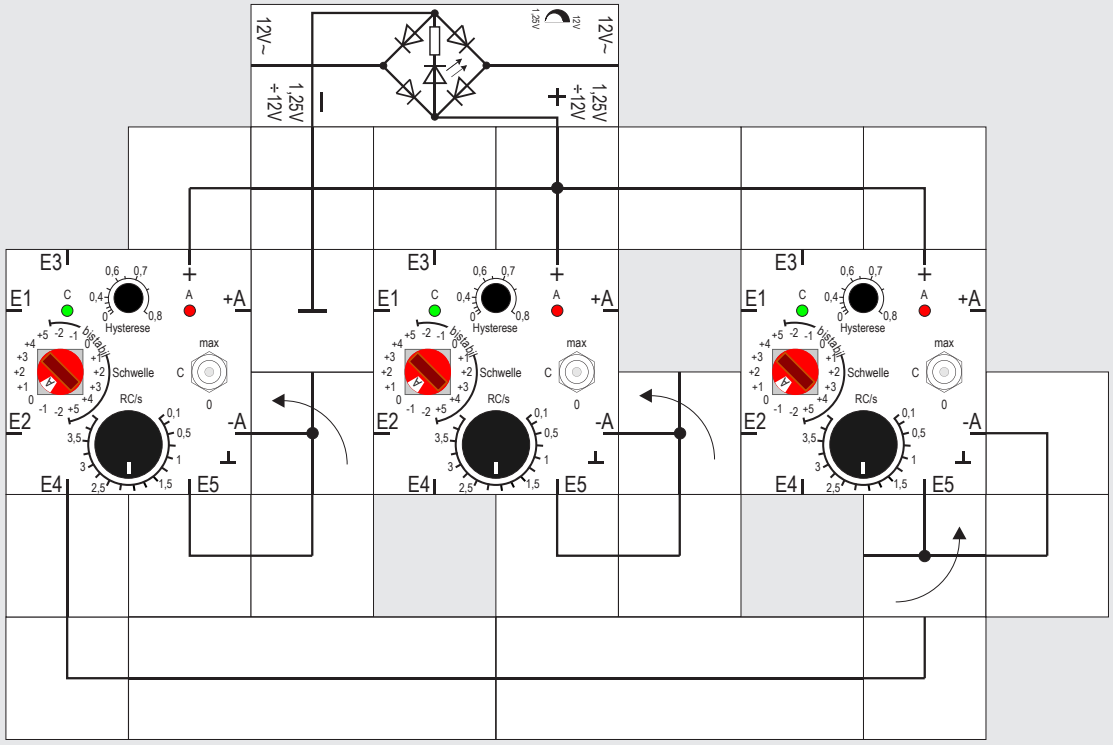
If we instead change the threshold of one of the modules from +1 to 0, this one turns on after a short time and soon the next one in a clockwise direction, while the first turns off. A pattern of one (sometimes two) turned-off modules »rotates« in the ring counterclockwise.

The lowering of the threshold of another module from +1 to 0 and maybe some fine-tuning do not change the picture. The fine-tuning might be necessary for a module not to be turned on permanently. The circuit does not show a stable state until all thresholds are 0. Then, all three modules are turned on permanently.


E3
E1
E2
E4
E5
+A
-A
C
A
Hysterese
bistabil
Schwelle
RC/s
max
12V~
1,25V
÷12V
12V
1,25V

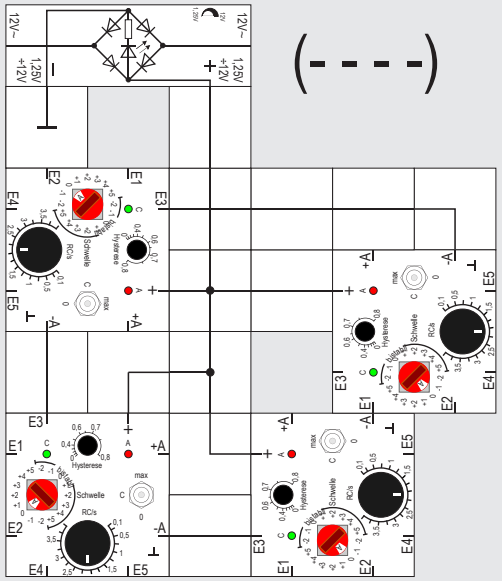
Experiment 20
Double coupling of three modules (===)

It is possible to couple three modules in a way that every module inhibits the other two, and again, all modules are equal.
If all modules have the same time constant and the threshold +1, nothing happens when we apply the supply voltage; all three remain turned off, which isn't surprising in this circle of very grumpy neighbors. Even if we turn all of them on as close to the same time as possible and release them simultaneously back to »C«, they are turned off in a little while.
If we decrease the threshold to 0 for all modules, we observe after applying the power that all green LEDs become brighter, but only one module is activated and turns off the other two.
This is very interesting! The circuit chooses a winner. And there is always only one. As we will see in further experiments in the biological area, this motif is used in real biology quite often: For example, it always happens when a cell determines its type and is fixed by a genetic circuit. We do appreciate that a nerve cell in our brain remains a nerve cell and does not randomly change to a muscle cell. Stabilizing such a state is really no problem for a circuit because of its multiple inhibiting couplings.

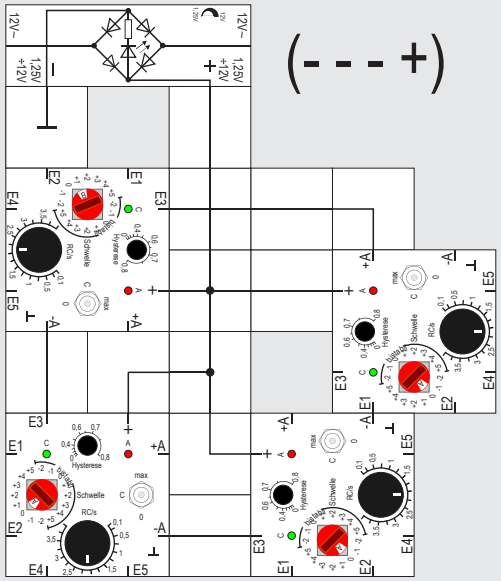

12V~
1,25V
+12V
E1
E2
E3
E4
E5
C
A
+A
-A
Hysterese
Schwelle
RC/s
max
bistabil
monostabil

Experiment 21
Coupling of three oscillators

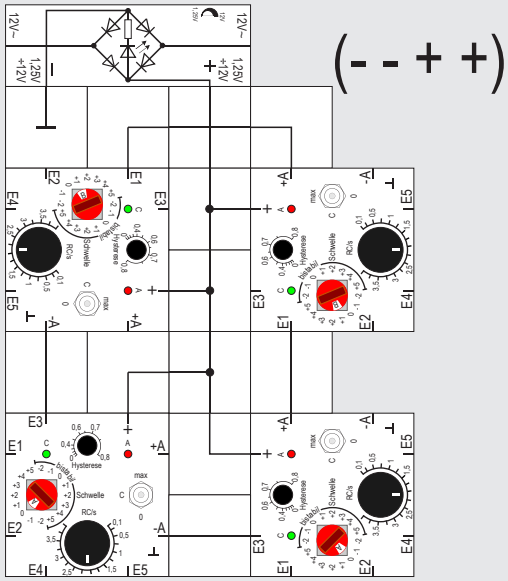
In experiments 12 and 13, we already coupled two oscillators in different ways and found out they synchronize to a certain extent. Before we apply the supply voltage to the present setup, we rotate the three T-connection modules 90° so the coupling of the oscillators is not yet active.
If we try to adjust the frequency of all oscillators with the time constants (RC) and hysteresis rotary knobs to the same value, we already know we won't succeed. If we watch the circuit for a while, we see that the phases of the oscillators drift apart, which is an inevitable effect of the slightly different frequencies of the oscillators.
Now, if we rotate the T-connection modules as shown in the circuit diagram and watch the oscillation picture after a few minutes, we will most likely find that all of them oscillate with the same frequency. One turned-on module »circles« and sometimes (maybe periodically?) small overlaps might be seen of two active modules (one LED lights before the previous one has turned off). It looks as if a fundamental oscillation and a superimposed oscillation attempt to change the phase shift. But it will always become stable again. In this experiment the oscillation picture depends a lot on the adjustments, and we can try lots of possibilities by cautious changes of the time constants. After every change we should wait a few minutes before trying to find a »continuous» new pattern.



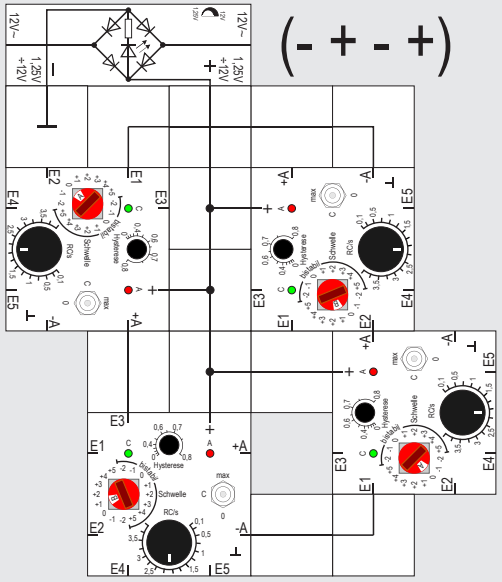
(- - - -)



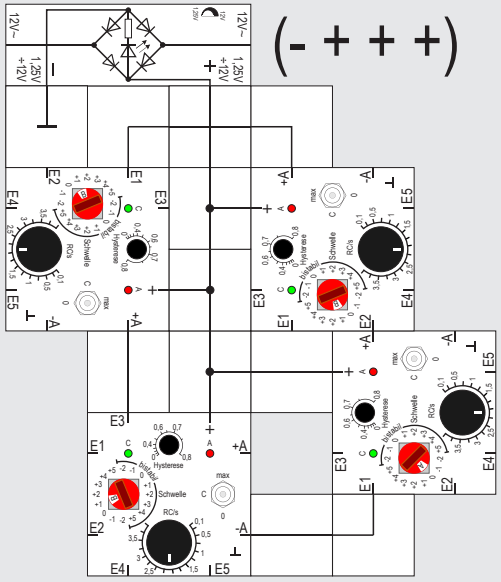
(- - - +)



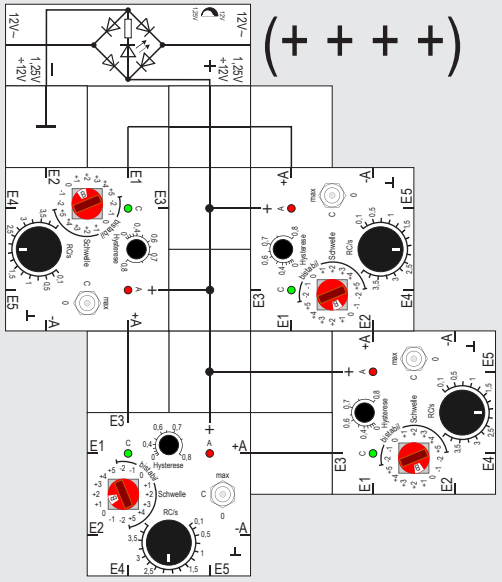
(- - + +)



(- + - +)



(- + + +)



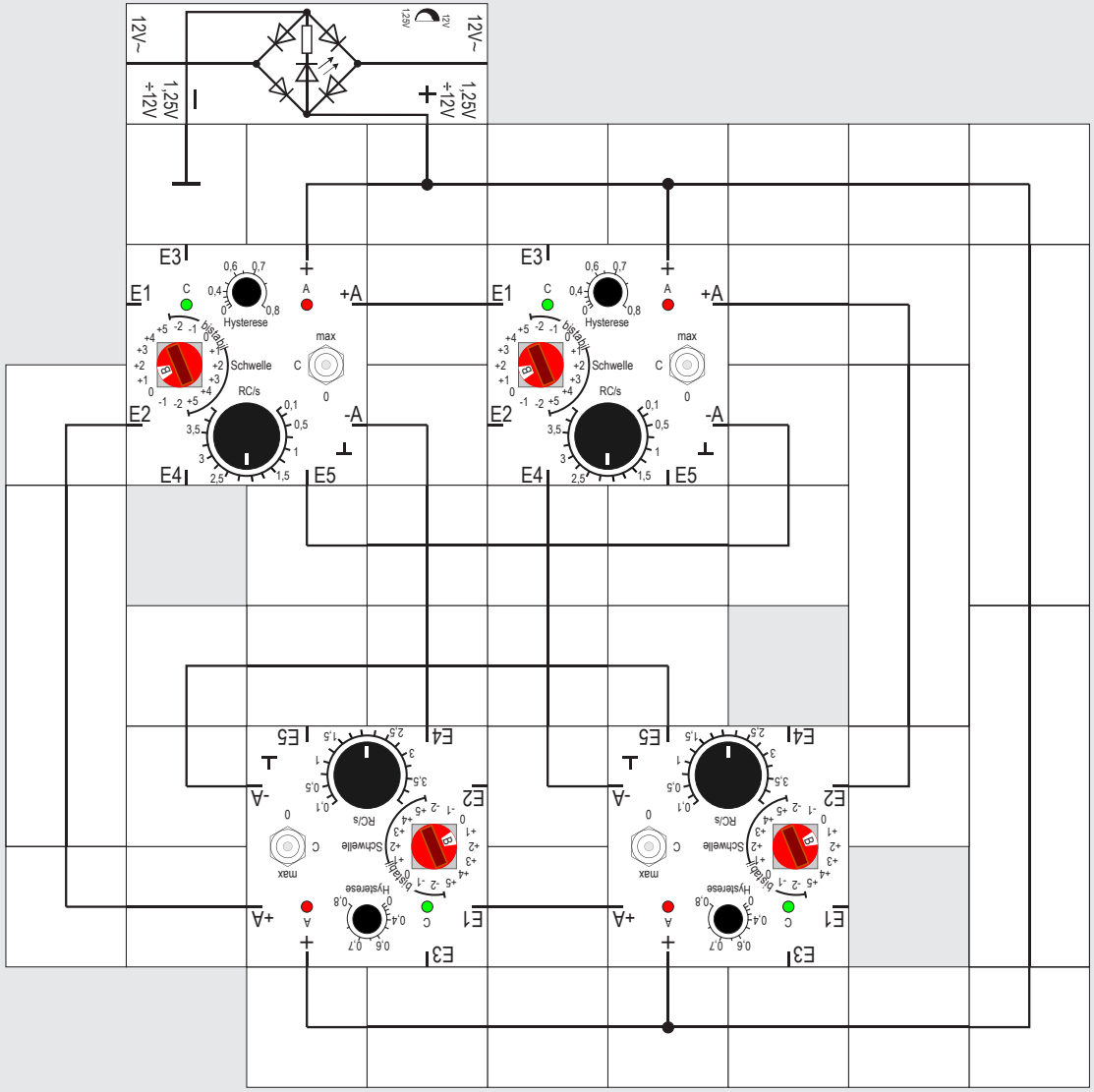
(+ + + +)

Experiment 22
Ring circuit with four gene modules

If we raise the number of threshold modules in our setup to four, there are even more possible coupling options, which are presented below in abbreviated form. Some configurations are predictable because of former experiments; others are surprising.

- (++++) This extension to experiment 18 gives a stable result: all four modules are turned off (all thresholds are +1). If we start a process with the toggle switch and the modules have the right setup (see experiment 28), we will observe a pattern for a few cycles. But we can notice a change: Either more and more modules turn on, which leads to a second stable state, or fewer do, and the starting state will result.
- (+++-) A permanent oscillation results if three modules are at the threshold +1 and the fourth at 0. This is the module connected to the -A output of the predecessor.
- (-+-+) Two modules are at the threshold +1 and the other ones, which are connected with the -A output of their predecessor, are at 0. This leads to a temporary oscillation; one of the two stable states is reached if one pair is turned on and the other one is off or the other way round. The set of pairs is predetermined because of the setup; therefore there are just two stable states.
- (--++) The thresholds are set up like before. One of the two stable states is reached if three modules are turned on and the last one is off. At the second state it is the other way around: The one that was turned off is now on and the other three are turned off. This depends, like in other experiments, on the time-settings. The same three modules are turned on or off; therefore, there are only two stable states.
- (---+) The threshold of one module connected to the +A output of its predecessor is +1; the other ones are at 0. This results in a permanent oscillation.
- (----) Beforehand you may consider that this is just an extension of experiment 15 with four elements, and there will be an oscillation if all thresholds are adjusted to 0; however, that does not occur. Even with a careful time adjustment, the modules become stable. A stable state is reached if every second module is turned on. The other two are off. There are two stable states; which modules are turned on and which are turned off depends on the previous time adjustment and the starting conditions. In general, an oscillation only occurs if you have an odd number of modules coupled in this way.

There are a lot of threshold adjustments besides the ones shown. Still, it should be clear that it is senseless in this ring circuit to choose a threshold such as +2 or higher, which can never be reached. It is the same if a threshold cannot be undershot. The same holds for the time adjustments: The number of possibilities is endless. We will arrive at »interesting« oscillations, even if they are not persistent, if we set the time values to similar values (see experiment 29). However, one thing is for sure: If there is any stable state, the circuit will arrive at it sooner or later.


12V~
1,25V
+12V
E1
E2
E3
E4
E5
C
A
+A
-A
Hysterese
Schwelle
RC/s
max
bistabil
monostabil

Experiment 23

Ring connection with four gene modules and double coupling (± ± ± ±)

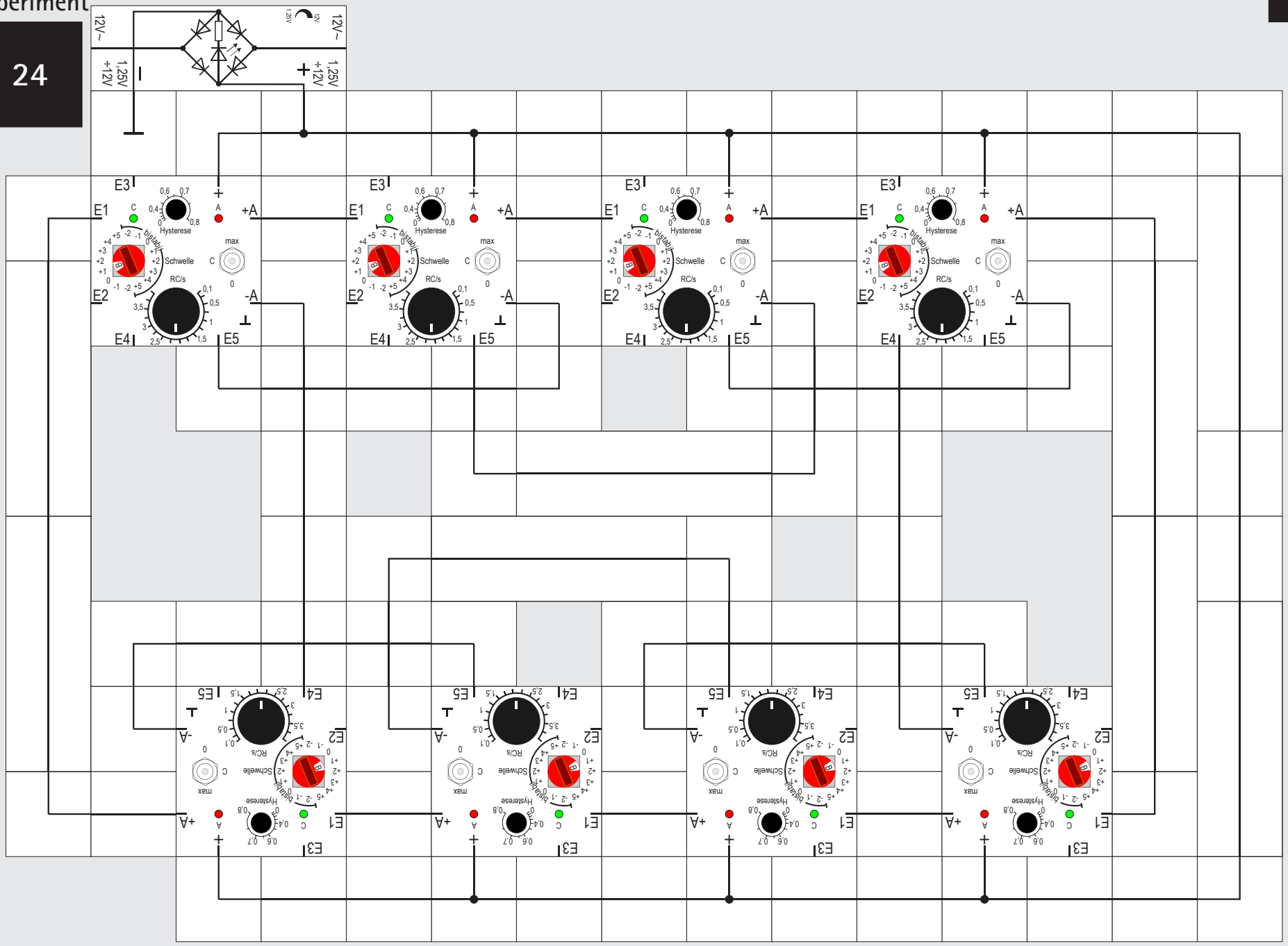
In some tries of experiment 22, you might have seen a stable state after some oscillation in the beginning. It isn't possible to adjust the times so perfectly that a module turned off just when its successor turned on. Therefore the circulating pattern becomes longer or shorter and eventually reaches a stable state.

If we couple four modules in a ring (++++), but at the same time make sure that little time shifts cannot be added randomly, we should be able to keep an oscillation alive. For this purpose we ensure that the predecessor in the ring will turn off in time by use of the -A output. This gives us the setup shown. All modules are equal and have a threshold of +1. The times of all four of them should be similar.

After applying the supply voltage, the oscillation doesn't start automatically; we have to launch it by a random toggle switch. Then a continuous, clockwise oscillation occurs, with two LEDs of adjacent modules lighting and the other two being dark. Our plan worked.

As the light advances from one module to another, it is possible that three LEDs light for an instant. It happens that only one LED lights, too. The counterclockwise coupling to turn off the predecessor seems to stabilize the oscillation.

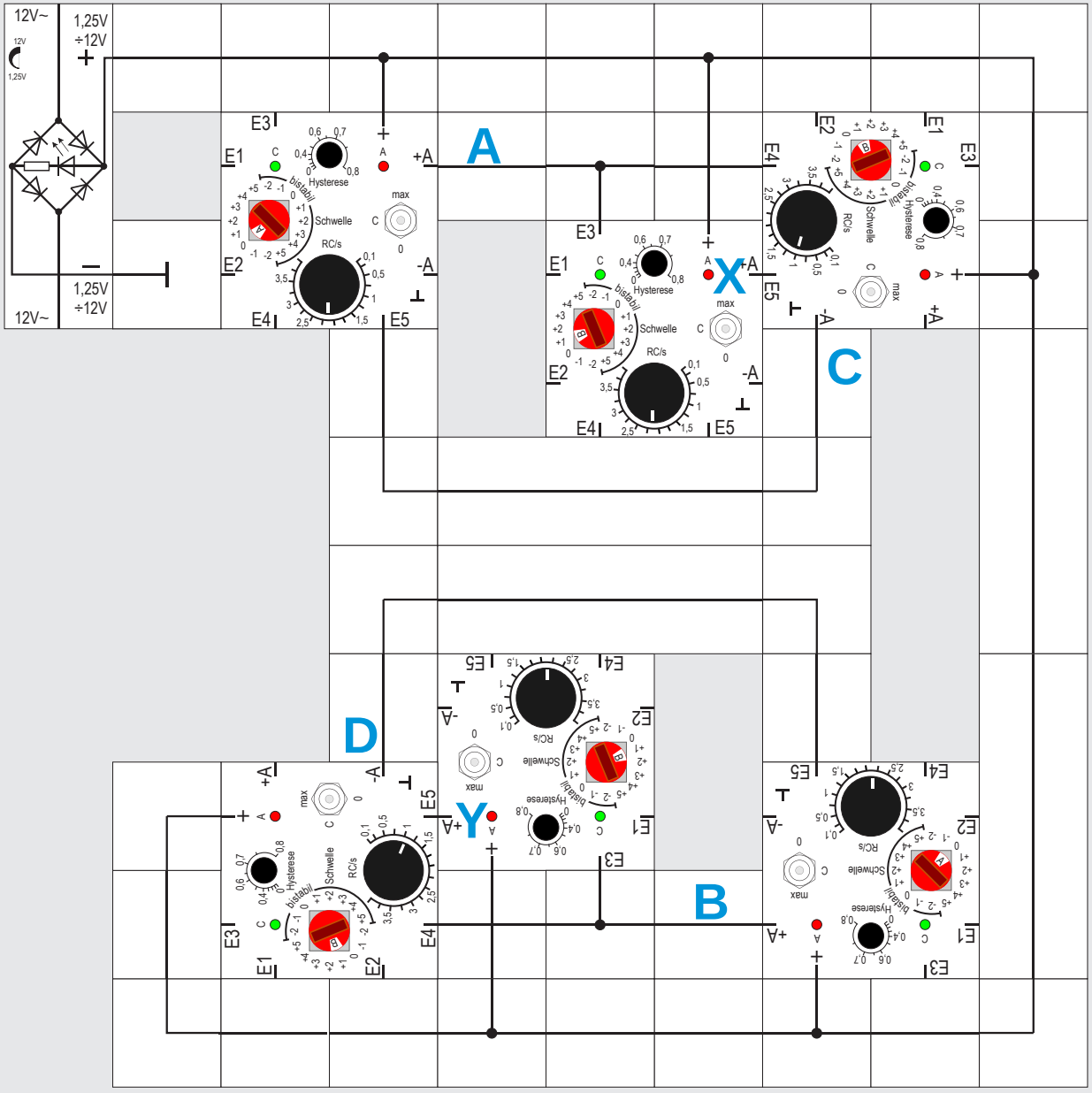
Now we are even able to change the time constant of a module quite a bit without bringing the oscillation to a standstill. So by this trick we can manage a stable oscillation in a ring connection with an even number of inhibiting couplings.


12V~
1,25V
+12V
E1
E2
E3
E4
E5
+A
-A
Hysterese
Schwelle
RC/s
max
C
A
B

Experiment 24

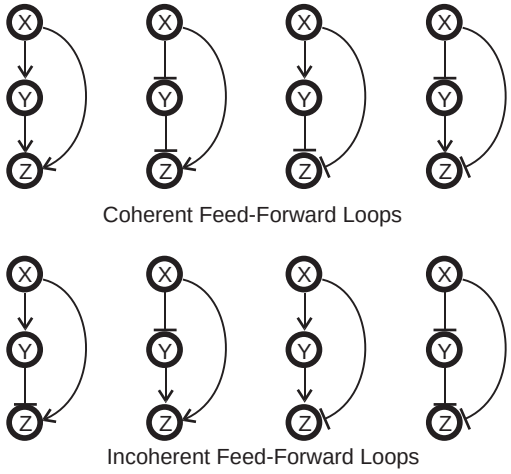
Orbital circuit with eight gene modules and double-coupling

We can extend the pattern that we found in experiment 23 to five, six or seven modules without changing it in principle. It is possible with some skill to add another circulating pattern to the already existing one, but not before we have installed an eighth module. To do this, we have to activate the module that has the biggest distance from the existing pattern by briefly operating its toggle switch. This will probably require some trial and error, but if you succeed, there will be two identical patterns circulating in the ring that keep the same temporal distance and therefore keep the oscillation alive. Four modules in the ring per circulating pattern are always enough to keep the needed distance between the adjacent patterns.


12V~
1,25V
÷12V
+
–
1,25V
÷12V
12V~
A
X
C
D
Y
B
+A
-A
E1
E2
E3
E4
E5
Hysterese
Schwelle
RC/s
bistabil
max

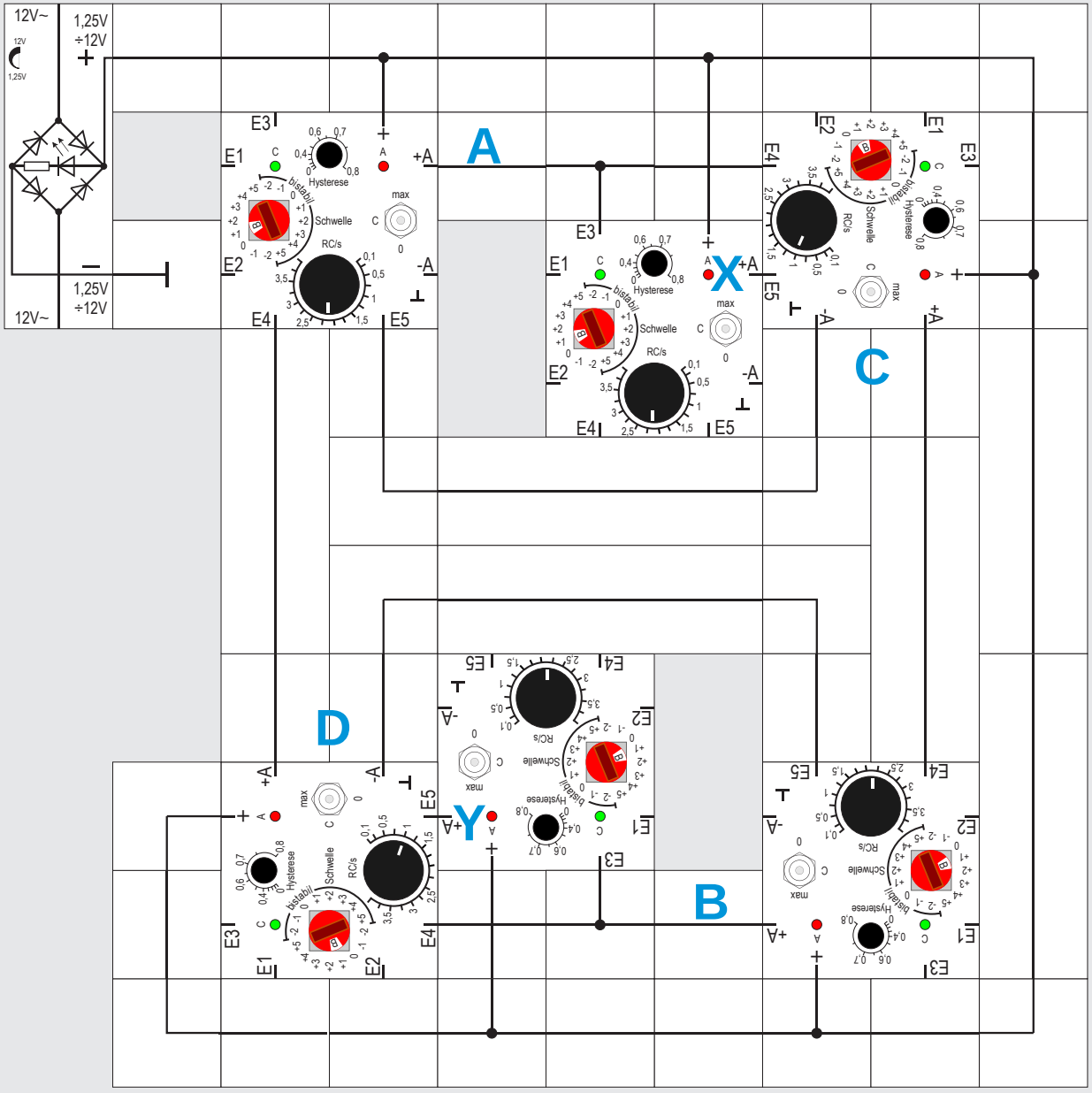
Experiment 25
Feed-forward loops

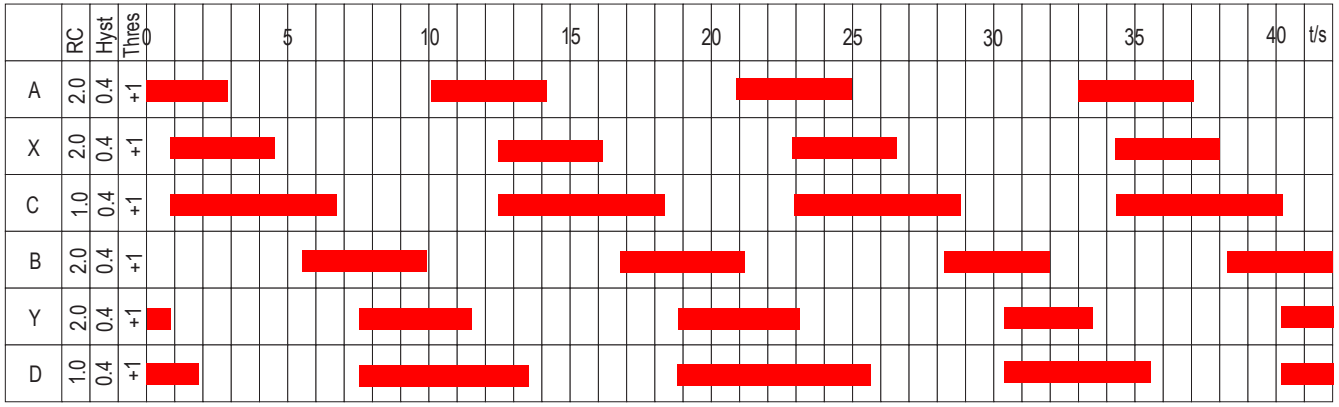
It was already noted in experiment 6 that in gene regulatory networks you will find small recurring structures in which gene X directly influences gene Z and indirectly influences it via gene Y. In analogy with musical compositions where subjects or motifs appear repeatedly, these structures are called motifs [10]. Since they are so frequent we can conclude that they have useful biological properties even though they are not obvious at first glance.
The influence of X on Z by two parallel pathways can be similar or dissimilar; so we call similar pathways »coherent feed-forward loops« and dissimilar pathways »incoherent feed-forward loops« (see figure). The effect of dissimilar pathways is easy to find out: Gene X activates gene Z and inhibits it again a little time later via Y, so gene Z was active only for a short time.
But coherent structures occur a lot more frequently than incoherent ones (85% compared with 15% for the well-known bacterium *E.coli*) and scientists have studied their possible use [11]. We want to comprehend this with an experiment using oscillators.
Our setup consists of two identical oscillators combined with a coherent structure. Gene A activates gene C directly and indirectly via gene X. When gene C is active it inhibits gene A. With thresholds of +1 for X and C and 0 for A, an oscillation occurs. The second, laterally reversed oscillator with genes B (threshold 0), Y and D (thresholds +1) does the same. The hysteresis of all modules is 0.4, with an RC of A, X, B and Y of 2s, and 1s for C and D.
After applying the supply voltage both oscillators vibrate with about the same frequency and a random phase shift.

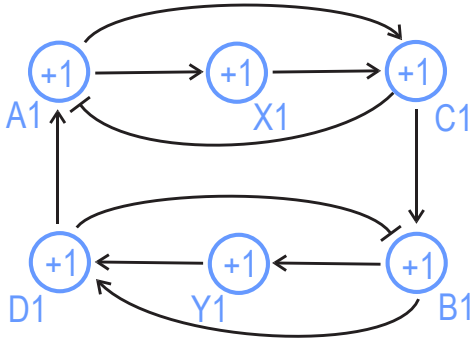


Experiment 26
Coupled oscillators

The behavior of both oscillators changes fundamentally if we couple them. Therefore the +A output of C has to be connected to the E4 input of B and accordingly the +A output of D with the E4 input of A. Furthermore the thresholds of A and B have to be increased to +1. We must start the oscillators now manually (preferably inversely phased) by using the toggle switch. It is sufficient if we put one switch for just long enough to the position »max« and then back again to »C«.
After a short observation period we find that both oscillators run simultaneously and are inversely phased. The phase opposition results from the active output signal of one oscillator inhibiting itself but activating the other. When for example gene C is active, gene D always turns off and the other way around (see diagram). This »active« state circulates clockwise in our experimental setup.
If we observe closely long enough, we notice that the phase of genes C and D is not constant but slowly fluctuates. For example, it is possible that both red lights are shining or both are off, and in the next cycle they are likewise on or off together. You may get the impression that the reason for this be-


12V~
1,25V
÷12V
12V~
1,25V
÷12V
A
X
C
D
Y
B
bistabil
Schwelle
RC/s
Hysterese
max



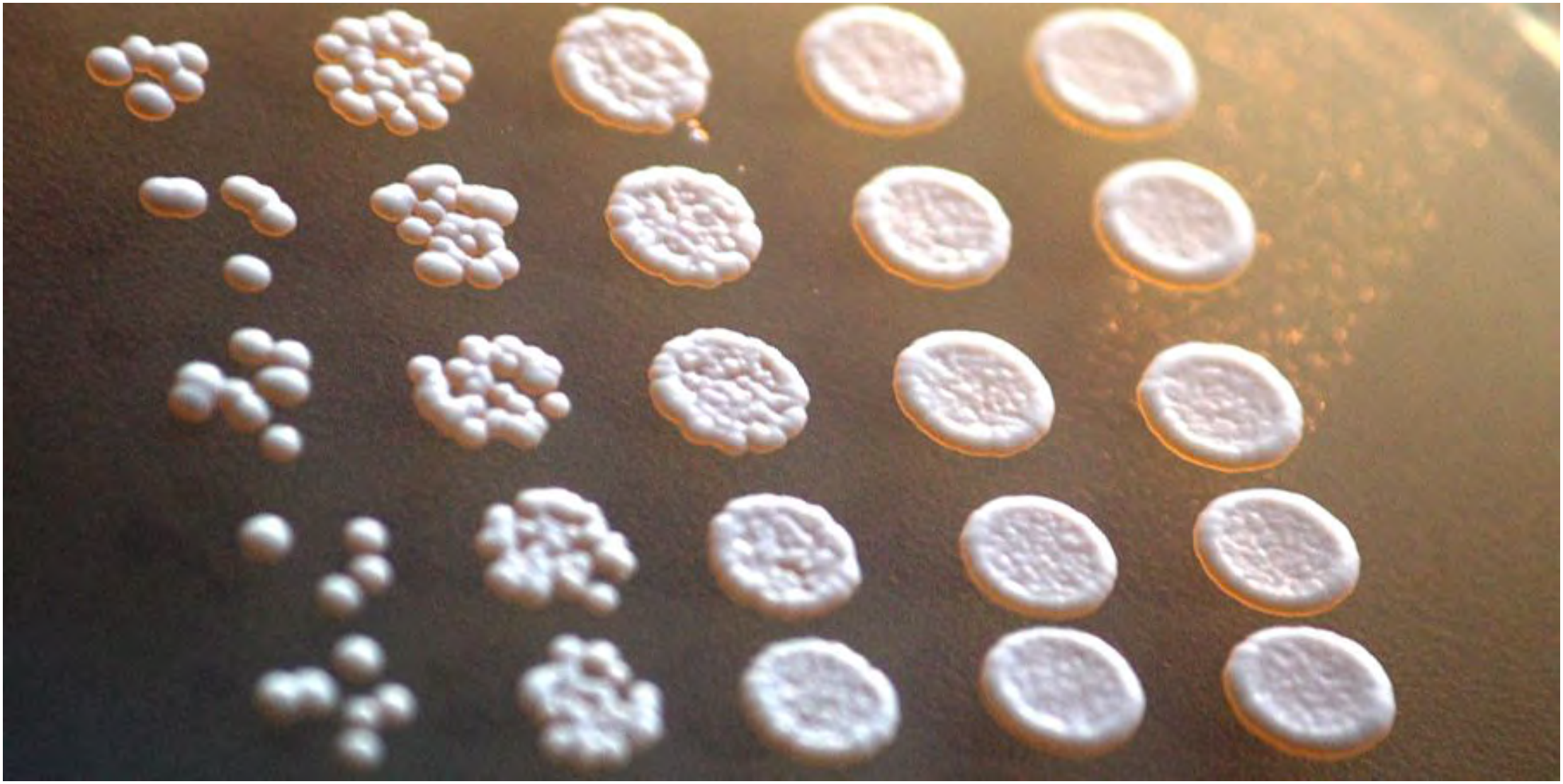


havior is the small tolerance of both signals causing a small drift away from each other, but they always get synchronized.

Now we go a step further and turn genes X and Y off manually with the help of their toggle switches, because it was probably not clear what they were actually good for. The turning off occurs best consecutively and when they are off anyway.

As expected, the pattern keeps on circulating clockwise, and therefore we may consider these genes dispensable. Curiously, in the previous experiment, we detected that a ring structure of this kind must have an odd number of cells; otherwise the oscillation won't be stabilized. And this is still true, because after some time it happens: All genes are turned off and nothing oscillates any longer. Transferred to the biological field, it is true that this type of coherent feed-forward loop is crucial for stability of coupling and for safety of information transfer. Therefore, it is no surprise when studies show that this is the most common network motif in biological regulatory networks.

Lectron



4. Gene networks of living cells

Parenthesis

In the past experiments we got to know the properties of the gene module and investigated the interaction of a number of modules. »But where is the biology?« you might ask. Fine, there has been a little circuit or two that worked to mimic something in a bacterium. But this LECTRON experimental kit is after all named »Gene Regulation« and hints at the regulation of vital processes.

Therefore we want to make a journey along gene networks of living cells and organisms in the following experiments. More precisely we will try to do something quite crazy with our modules: We will rebuild real regulatory networks molecular biologists have found in living cells! Of course, we do not expect our simulation to come to life. Our lack of knowledge about living matter is far too big, and matters and questions related to artificial life largely belong to philosophy, anyway. The whole regulatory network of cells is enormous and still mostly unknown: Only very small portions of it are thoroughly known. However, these small, well-known portions are indeed excellent for studying the characteristics of how the functions of cells are controlled. For this experimental kit we chose real treasures of these small networks that are known in detail. They control very different phenomena of life like cell division, the early embryonic development of a multicellular organism, even metamorphosis of different cell types, in addition to differentiation of stem cells and the recently discovered possibility of reprogramming differentiated cells into stem cells.

Our first example will be a gene circuit controlling the cell division of the common baker's or brewer's yeast, so it is not an insignificant process. The control task is more complicated than the previously built circuits and is about as complex as the program of a modern washing machine or a dishwasher. Therefore, yeast cells at least have to have the intelligence of a washing machine to reproduce themselves. Well, we don't want to discuss whether the actions of a cell can be called intelligent. However, we do want to see the information processing in a yeast cell in action and therefore construct our first yeast network in the following experiment with gene modules.

Experiment 27
Simulation model of baker's yeast

In this experiment we want to build part of the biochemical regulation network of baker's (also known as brewer's) yeast (*Saccharomyces cerevisiae*) that is responsible for the control of cell division, and thus the reproduction of the cell [12]. Yeast live all around us. Wherever there are fermentable, sugar-rich juices, you can find many different kinds of yeast. They belong to the family of fungi, and their cells have a nucleus, so they are called »eukaryotes«. Due to their small genome, their small number of chromosomes and the short period for cell duplication, baker's yeast has become a model organism for scientists doing research in molecular genetics. Yeast was the first eukaryote whose genome was completely decoded within the scope of the Human Genome Project. The propagation cycle of baker's yeast is very well known and often serves as a textbook example of cell division.

Its control process is well-known, too; it is a small network of chemical reactions playing the role of the clock signal for the different phases of cell division. This biochemical network of proteins and their genes can even be modeled mathematically very precisely by coupled differential equations predicting the time course of the concentrations of the biochemical players. These academic depths, however, are not what we want to explore with our model. On the contrary, we take advantage of the knowledge that a protein's concentrations usually are very high or close to zero and that the change between these states happens quickly. This sounds like we could express these states as binary states, 0 or 1, doesn't it? If we then confine our attention to the most important proteins and combine those with similar dynamics into one node, we get a network consisting of eleven proteins and the corresponding genes modeled as binary nodes with either of two states, 0 or 1. The state of each gene depends on the input signals it receives from the proteins of other genes. We neglect the details of the biochemical strength of reaction (because of the protein's concentrations often being at extreme levels) and only distinguish between activating (+) and inhibiting signals (-) that can at best be weighted with small integers like 1 or 2. The signals are sent in a defined direction of signals illustrated by an arrow.

The dynamic behavior of the network, meaning the blinking pattern of the gene activities, is determined for every gene; the chronological next step is a result of the signals it receives from other genes. The dynamic rules can be summarized as follows: Every node S_i has two different possible states, $S_i = 0$ (inactive) or $S_i = 1$ (active). When a node is active it can affect the nodes that are connected to it. These connections can be activating ($a_{ij} > 0$) or inhibiting ($a_{ij} < 0$). Every target node j adds the weights a_{ij} with the correct sign and reacts after a little delay with either activity or inactivity, depending on if the sum Σa_{ij} reaches its adjustable threshold Th_j or remains below it. This might sound a little complicated, but it is actually easy, as you can see in this example (sketch on the following page):

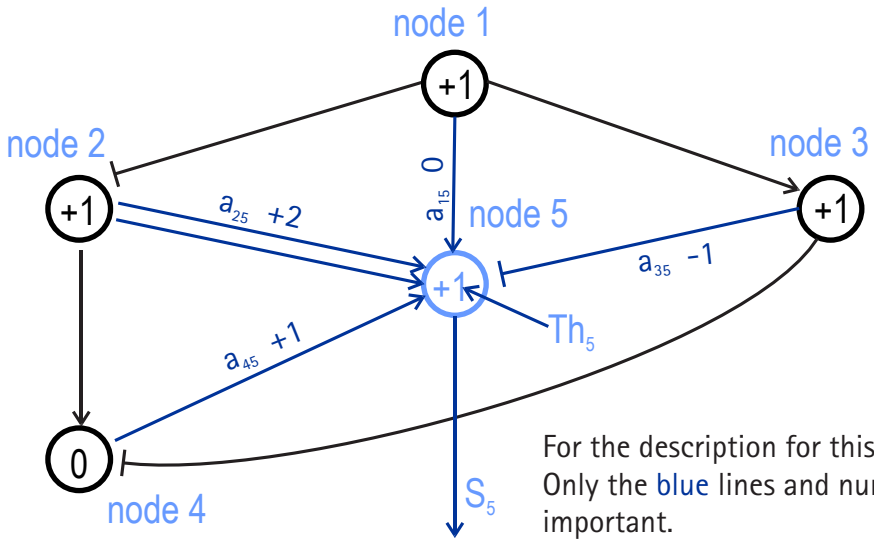
At time t, the target node S_5 (j = 5) with threshold $Th_5 = +1$ that is connected to nodes S_1 to S_4 (i = 1, 2, 3, 4), receives the following weighted input signals: $a_{15} = 0$, $a_{25} = +2$, $a_{35} = -1$, $a_{45} = +1$. If the sum (i=1...4) $\Sigma a_{i5} = 0 + 2 - 1 + 1 = 2$ is larger than $Th_5 = +1$, the target node S_5 will be activated (or remains activated, if it was) and sends itself the output signal $S_5 = 1$ after delay Δt at time $t + \Delta t$, which can be weighted to activate or inactivate other nodes.

If the threshold Th_5 had not been reached because it was, for example, $a_{25} = 0$, the node S_5 would have been deactivated at time $t + \Delta t$ ($S_5 = 0$) and could not have influenced other nodes for the time being.

The current states (time t) of all nodes sending signals to node S_5 determine its state in the next time step, $t + \Delta t$.

Concerning genes, there are two different types that differ in how they react if they do not get an input signal anymore, so the sum of their input signals is 0. One type keeps sending its signal; the other is inactivated. The first behavior is regarded in our gene modules by using bistable thresholds, the second by normal thresholds.

Note that there is no central clock signal; the behavior of the whole network depends only on the activating and inhibiting connections between genes, the thresholds and the chosen time delay Dt (RC and hysteresis).



For the description for this example, see text. Only the blue lines and numerals are important.

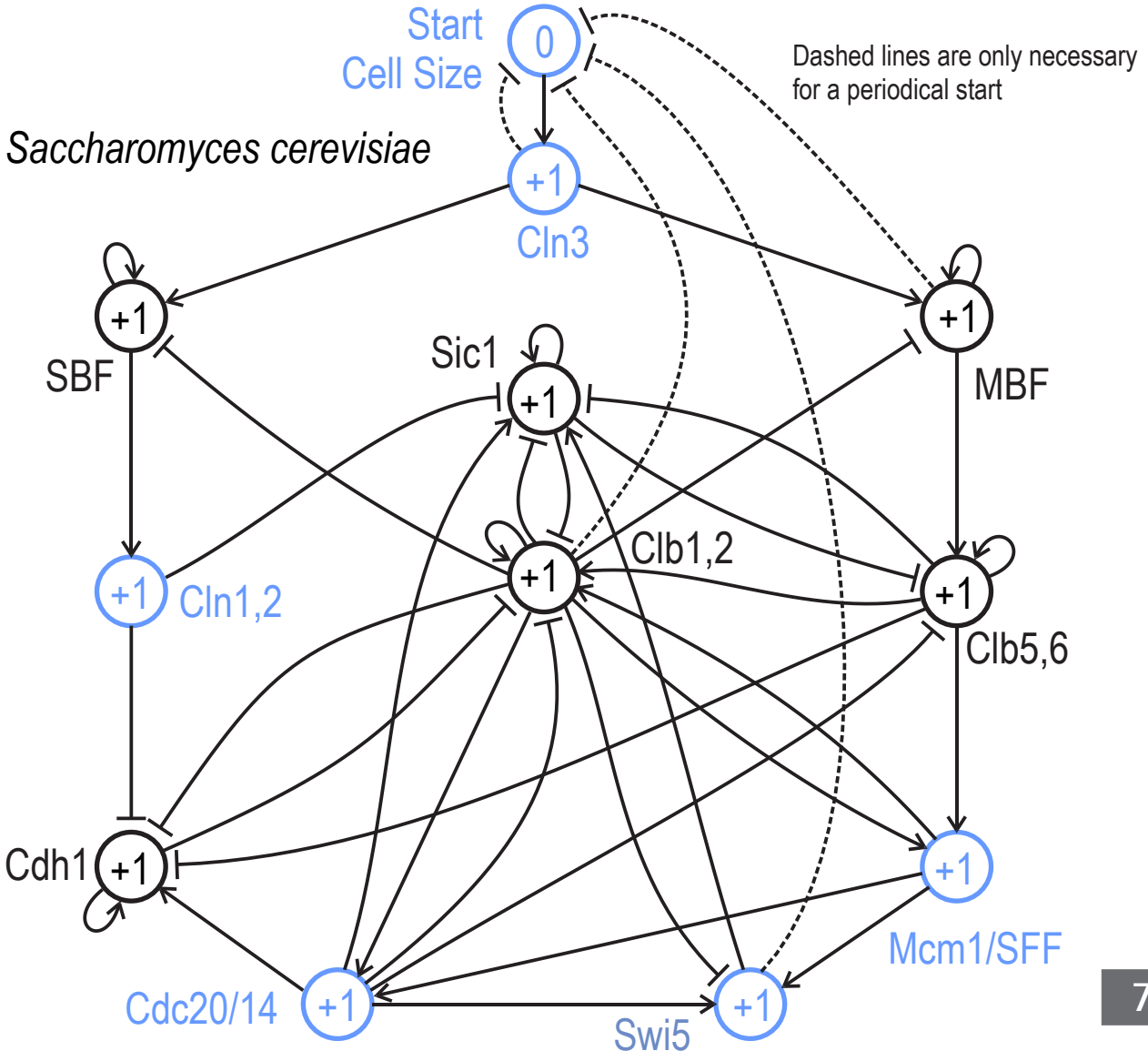
Our yeast model at first consists of, as we already said, 11 nodes representing about 20 key genes of the baker's yeast network. By this, we display the heart of the gene regulatory network. Every node is connected to certain other nodes to express possible effects between the corresponding genes and proteins. In the following table the genes are listed with their names, like **Cln3**, **SBF** and **MBF,** on the left and the top, and their connections below.

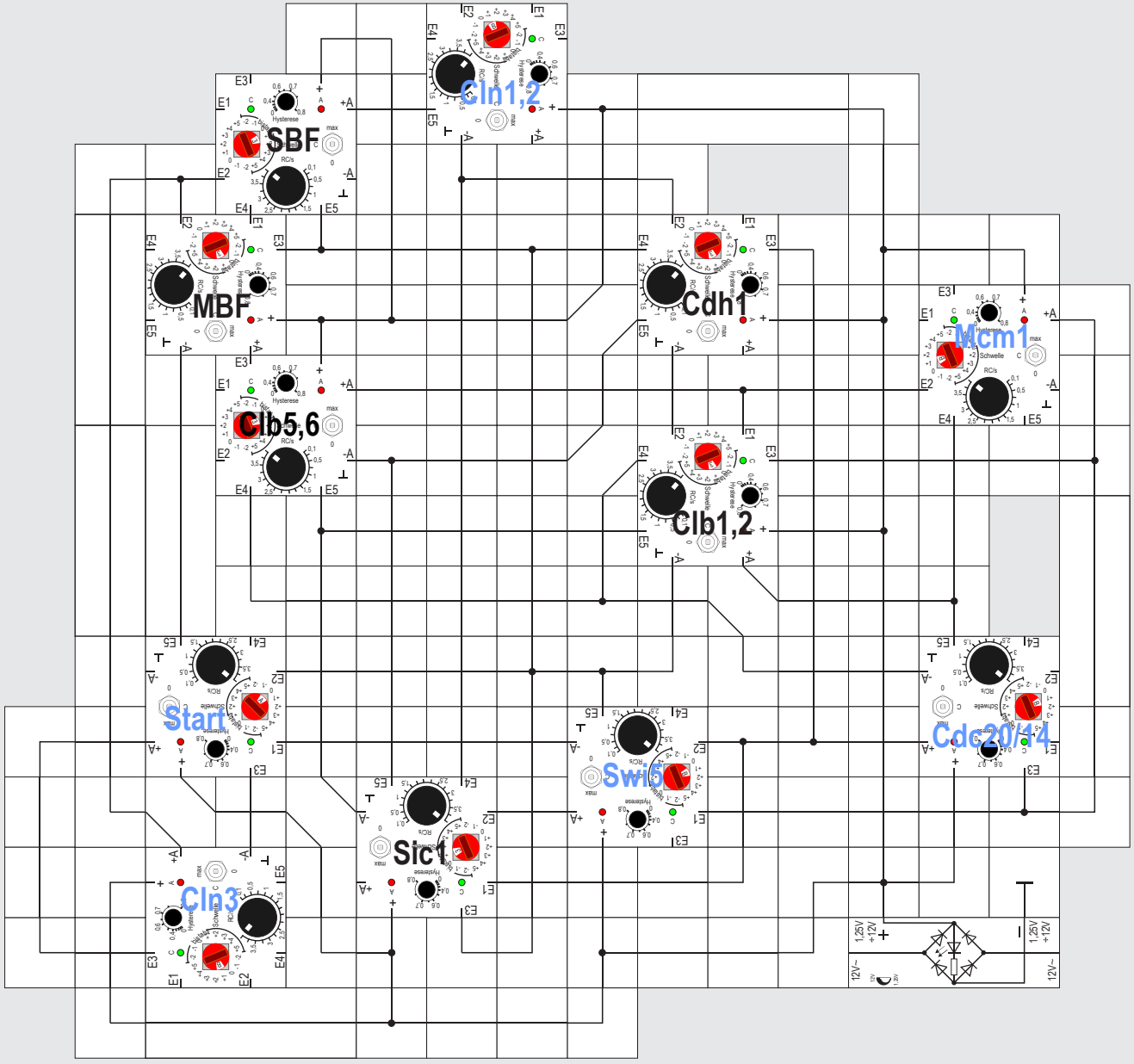
to ↓ from→	Start	Cln3	SBF	MBF	Cln1,2	Sic1	Clb5,6	Cdh1	Clb1,2	Mcm1	Cdc20/14	Swi5	Threshold
Start	0	-1	0	-1	0	0	0	0	-1	0	0	-1	0
Cln3	**+1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**+1**
SBF	**0**	**+1**	**+1**	**0**	**0**	**0**	**0**	**0**	**-1**	**0**	**0**	**0**	**+1b**
MBF	**0**	**+1**	**0**	**+1**	**0**	**0**	**0**	**0**	**-1**	**0**	**0**	**0**	**+1b**
Cln1,2	**0**	**0**	**+1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**+1**
Sic1	**0**	**0**	**0**	**0**	**-1**	**+1**	**-1**	**0**	**-1**	**0**	**+1**	**+1**	**+1b**
Clb5,6	**0**	**0**	**0**	**+1**	**0**	**-1**	**+1**	**0**	**0**	**0**	**-1**	**0**	**+1b**
Cdh1	**0**	**0**	**0**	**0**	**-1**	**0**	**-1**	**+1**	**-1**	**0**	**+1**	**0**	**+1b**
Clb1,2	**0**	**0**	**0**	**0**	**0**	**-1**	**+1**	**-1**	**+1**	**+1**	**-1**	**0**	**+1b**
Mcm1	**0**	**0**	**0**	**0**	**0**	**0**	**+1**	**0**	**+1**	**0**	**0**	**0**	**+1**
Cdc20/14	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**+1**	**+1**	**0**	**0**	**+1**
Swi5	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**-1**	**+1**	**+1**	**0**	**+1**

Admittedly hundreds of genes are involved in cell division, but only these 11 are responsible for the central regulation and control. In the table there is an additional »gene« listed, named **Start**, whose only purpose is to start a new run automatically after a complete run of the simulation is finished. We don't have to initiate it manually; it runs periodically. In nature, this start signal is triggered when the cell has reached a sufficient size.

In the table, -1 in a column means an inhibiting signal with the weight 1, and +1 is an activating signal with the weight 1. At first there are no other weights than 1 in this example. The last column, »threshold«, lists the corresponding threshold of the gene that has to be reached by the sum of all input signals to activate the gene. Except for the Start gene, all thresholds are +1; the additional **b** means **bistable**.

We find the list of connections of the setup in the table above. In order to make it easier to view, it is displayed in a figure on the right.
The circles are the single genes and their names; the number in the circle is their threshold. The direction of signals is marked with an arrow for activating signals. Inhibiting signals have a crossbar at the recipient's side instead of the arrowhead. Bistable genes have an additional loop and are marked in black; normal ones are displayed in blue.
An example for the connections, last one Swi5 of the table, goes like this:
Swi5 has the threshold +1, so it has a blue circle with a +1; it receives an inhibiting signal from **Clb1,2** and an activating signal from both **Mcm1** and **Cdc20/14**. It itself sends an inhibiting signal to **Start** and an activating signal to **Sic1**. Both signals are listed in the next-to-last column with -1 and +1 below **Swi5**.
A possible compact setup with gene modules already adjusted to the same time delays (hysteresis = 0.8, RC = 4.5s) and correct thresholds based on this figure is displayed on the next page.




Cln1,2
SBF
MBF
Cdh1
Mcm1
Clb5,6
Clb1,2
Start
Swi5
Cdc20/14
Sic1
Cln3
1.25V
+12V
12V~

Before we apply the supply voltage module, we double-check that our setup is correct. Under no circumstances can the outputs +A and -A be connected to the supply voltage or to each other; otherwise irreparable damage will occur to the gene modules. Also, it is good to check that all toggle switches are in position »C«; otherwise the desired process probably won't work. And here is a last preliminary remark: In a big setup like this, the magnet at the bottom of the supply voltage module that is also used as a heat sink becomes quite warm. After applying the supply voltage, the simulation of the cell division cycle should start itself. First, the green LED of the **Start** gene module turns on, a little later its red one, and as a consequence **Cln3** will be activated. We can also see this easily in the connection diagram. There we find that next **SBF** and **MBF** are activated while **Start** will be turned off. After this it becomes difficult to predict what will happening only by looking at the connection diagram. The table on the left will help us along. It indicates with 1, which genes are active, and with 0, which are inactive. Changes in a time step are marked in red.

The additional gene module **Start** is in the table, too, but distinguished from the others by the use of thin letters.

In the very first run **Cdh1** and **Sic1** are not yet 1, so are inactive. We can change that by switching the three-position toggle switch to »max« for a moment and then back to »C«, but we don't have to. The simulation starts anyway and in the second run, everything acts like the table predicts.

At this point, a tip: If the simulation runs differently or not at all, debugging is inevitable. Probably the wiring is incorrect and has to be carefully checked. We can see whether all modules are connected with the supply voltage if we switch to the »max« position. All red LEDs should light. Don't forget to switch back to the »C« position. In this table we do not see how long the gene module remains in its active or passive phase. We find that out if we, for instance, film it with a digital camera and at the end, note the times when the red LED of each gene module started or stopped being lit. Then we get a diagram like the one on the next page.

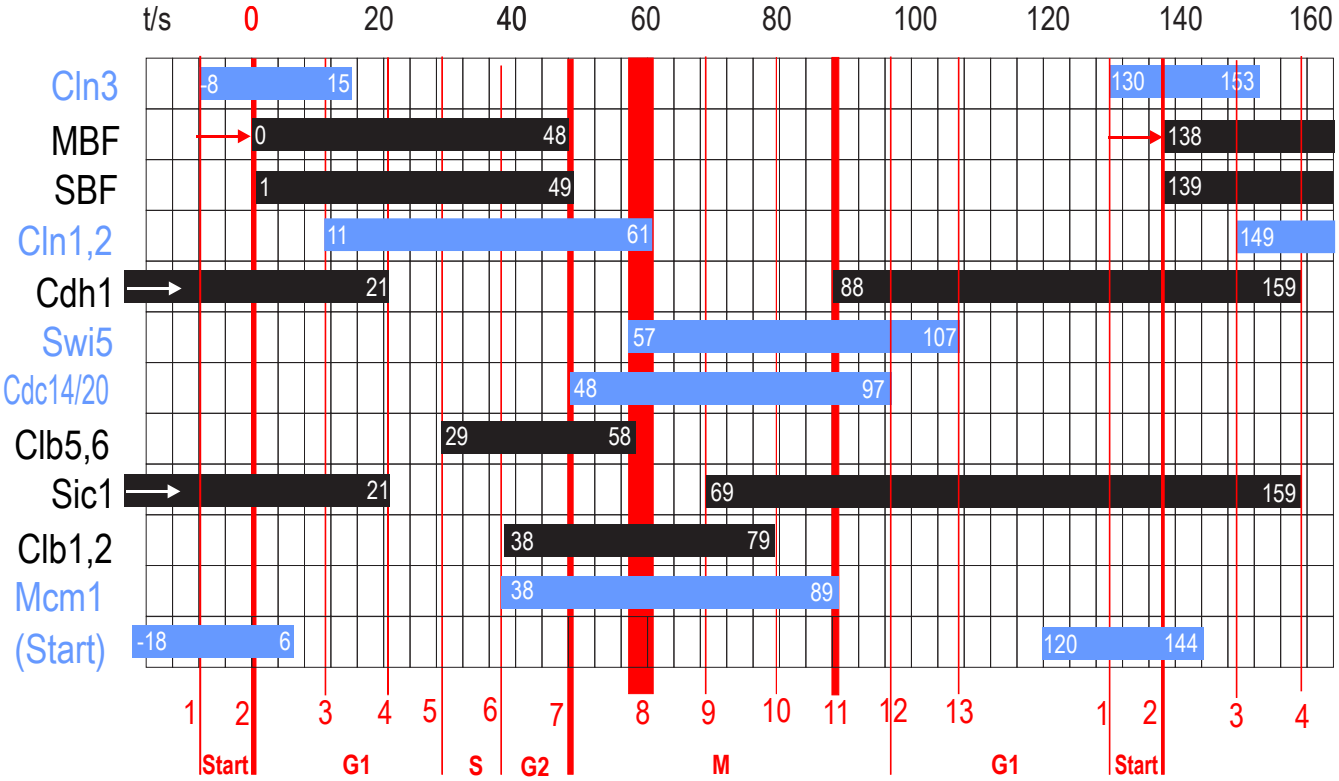
Step	Start	Cln3	MBF	SBF	Cln1,2	Cdh1	Swi5	Cdc20/14	Clb5,6	Sic1	Clb1,2	Mcm1	Phase
0	1	0	0	0	0	1	0	0	0	1	0	0	Start
1	1	1	0	0	0	1	0	0	0	1	0	0	Start
2	0	0	1	1	0	1	0	0	0	1	0	0	G1
3	0	0	1	1	1	1	0	0	0	1	0	0	G1
4	0	0	1	1	1	0	0	0	0	0	0	0	G1
5	0	0	1	1	1	0	0	0	1	0	0	0	S
6	0	0	1	1	1	0	0	0	1	0	1	1	G2
7	0	0	0	0	1	0	0	1	1	0	1	1	M
8	0	0	0	0	0	0	1	1	0	0	1	1	M
9	0	0	0	0	0	0	1	1	0	1	1	1	M
10	0	0	0	0	0	0	1	1	0	1	0	1	M
11	0	0	0	0	0	1	1	1	0	1	0	0	M
12	0	0	0	0	0	1	1	0	0	1	0	0	G1
13	0	0	0	0	0	1	0	0	0	1	0	0	G1

The diagram on the right shows a possible temporal progression of the simulation with the periodic start. As the time scale on the upper side shows, a complete run takes 138 seconds. The time steps from the table are marked with red bars. They have different widths because in our first RC adjustment of our modules, some don't change their activity at the same time. Especially in step 8, when this happens, a transition area results. By a new fine-tuning (careful changing of RC and hysteresis) we can try to minimize these areas.

The last column of the table above tells us four phases, G1, S, G2, and M, which run in this order in the cell cycle of all eukaryotic cells:

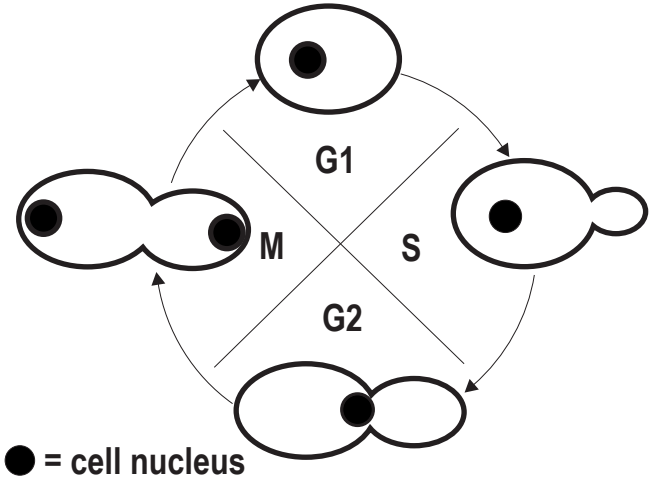
The doubling of the DNA in the synthesis or S-phase is temporally separated from the distribution of the DNA to the daughter cells, which happens in the mitosis or M-phase. These two phases of the cycle are separated from each other by interposed intermediate phases called gap phases. The gap phases G1 and G2 enable a control to ensure the previous phase of the cell cycle has been completed correctly and that the genetic information has not been damaged. Additionally the cell decides in the first gap phase G1 if it begins another cell division.

We can see from the time diagram which gene is active in which phase. To make the connection

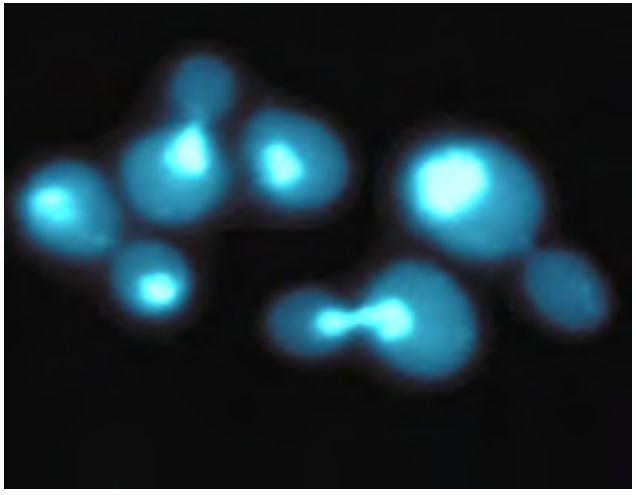


blue self-reducing
black bistable
all thresholds +1(except Start = 0); Hyst = 0.8 ; RC = 4.5s
period 138s

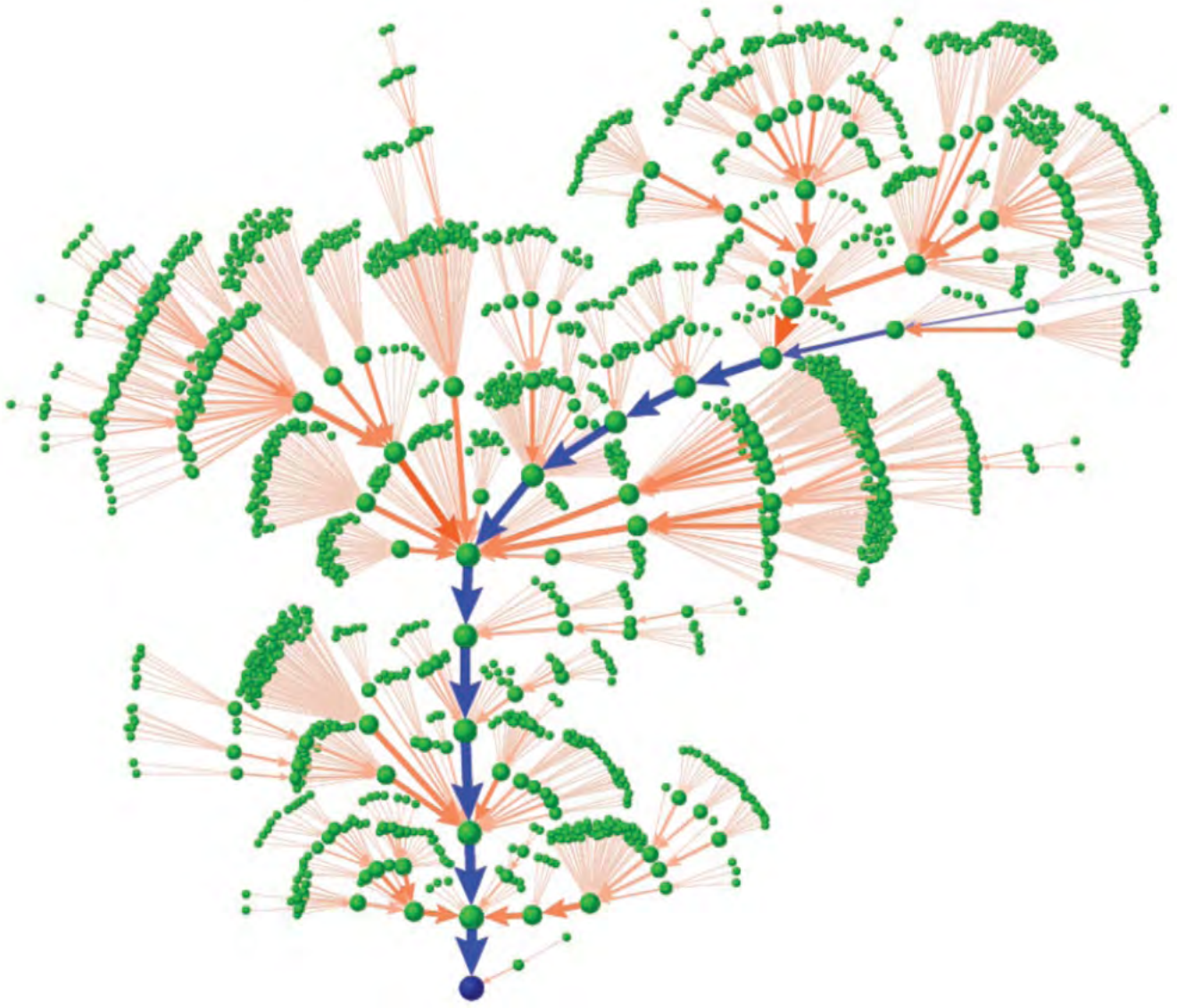
G1: The cell is growing and starts its division.
S: DNA is produced and the chromosomes are copied.
G2: Gap phase.
M Chromosomes are separated and the cell divides,
the process of mitosis takes place.
At the end of the M phase the cell starts the G1 phase and the cycle is completed.
G1, S, G2 form the so-called intermediate phase.



between simulation and reality clear, there are sketches and photos of the cell in the four phases.
When the simulation runs as desired and we are familiarized with the usual course, we can make modifications to the simulation model. Changing the time adjustments usually causes longer transition areas, but they can disturb the course seriously, too.
But we can interfere even more and turn single genes off; in a real cell this would be a mutation. If we switch the corresponding three-position switch



to »0«, this is equivalent to a so-called knockout mutant. Transferred to the biological process, this means that the protein is no longer produced. We can also simulate the opposite, the permanent production of a protein. Just switch one of the genes to the »max« position.
Mutants like this and even multiple mutants are well-studied, and it is known which mutants survive and which mutations stop cell division, resulting in a cell that is unable to reproduce itself. In our setup it is easy to find examples for each of these.



The attractor in the simulation model

For the simulation, we used 11 gene modules if we don't count the **Start** module. That one came into play only because we wanted the simulation to be periodic. If we shut it down (switch it to »0«), the simulation can be started with the **Cln3** module manually. Switch it to the »max« position for a moment. The usual simulation runs once. But we can also start the simulation with different initial conditions.

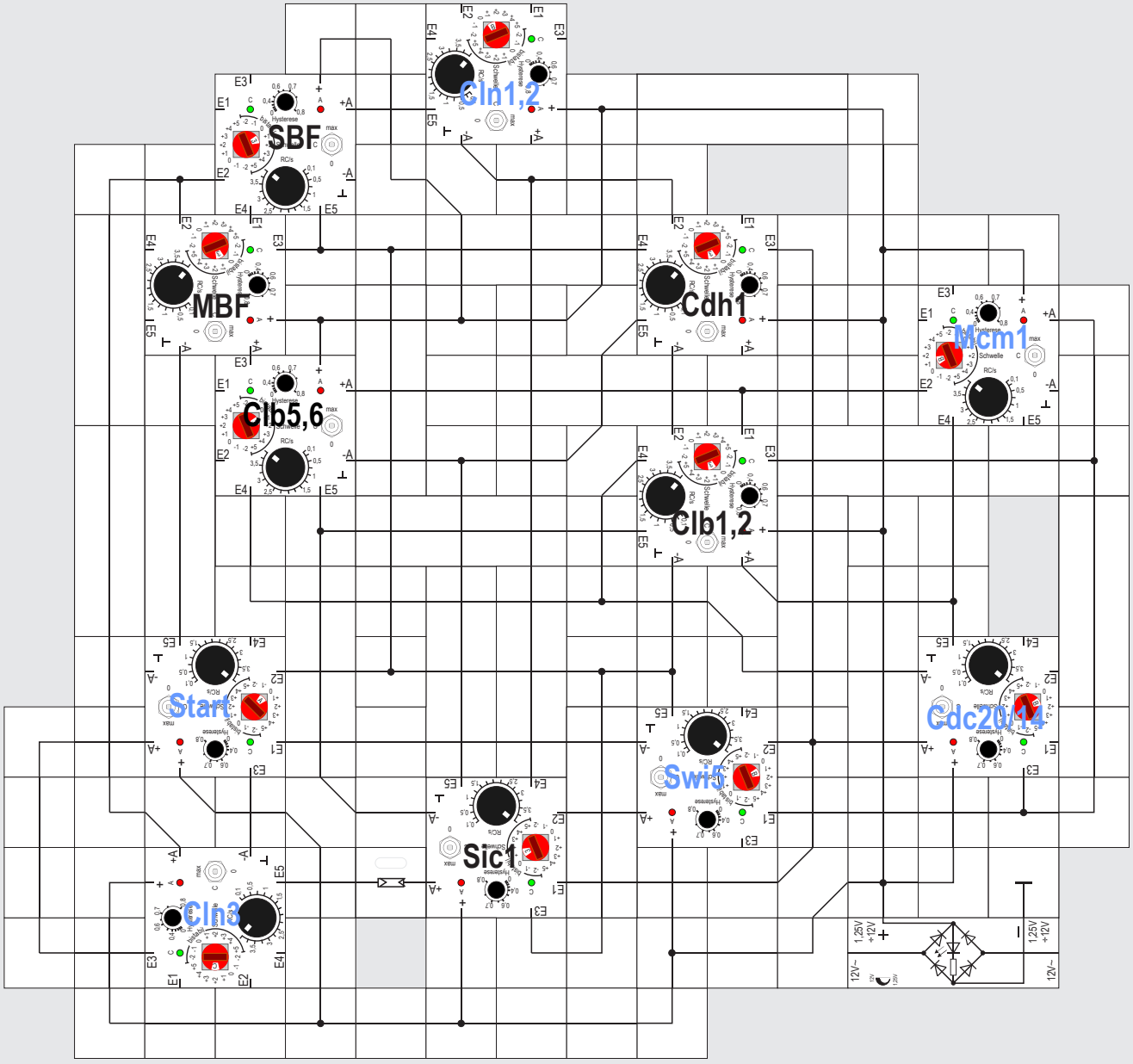
Since we used 11 modules and each of them has two states (active and inactive), we could display $2^{11} = 2048$ different states with the modules of our setup. The red LED shows if a module is active or not.

In a usual run of the simulation we find only 13 of these states, as listed in the table. If we set all modules by hand and quickly switch all of them back to »C« position, the simulation starts from this randomly chosen initial situation. If we did this with all of the possible 2048 initial situations and noted the respective time courses in detail, we would find a very interesting result: 1764 (86%) of them lead over intermediate states to one of the 13 states we saw in the usual sequence. In the figure on the previous page, this sequence tree is displayed [12]. Each circle is one of the 1764 states; the 13 small blue circles are connected by blue arrows forming the so-called attractor. Once the attractor is reached, it runs to the end, so to G1 phase (blue point on the bottom). If we bring it back to the automatic start, it will be almost irrelevant in which state the network starts. After one run, the sequence of states will follow the attractor, and it will not be left again in the following runs. So if the network jumps to a state outside the attractor due to a disturbance, the chance will be high that its course leads back to it and the next cycle is the usual again.

With regard to the modeled biological process, this means that the division's cell cycle is robust against disturbances.

Of course you will wonder if the appearance of such a strong attractor in a network of this size might be a coincidence or perhaps, for example, a result of evolution. Simulations with 1000 random networks of the same complexity, same number of connections, and same gene module adjustments showed that the typical attractor is smaller. The average size is about 40% (compared with 86%). This could be an indication that the size of the biological attractor is optimized to get additional dynamic robustness.

Here, our considerations about the simulation network of baker's yeast come to an end and we continue with a small comment on how to control our circuits with light and a useful way of fine-tuning our modules before building the network of another yeast.

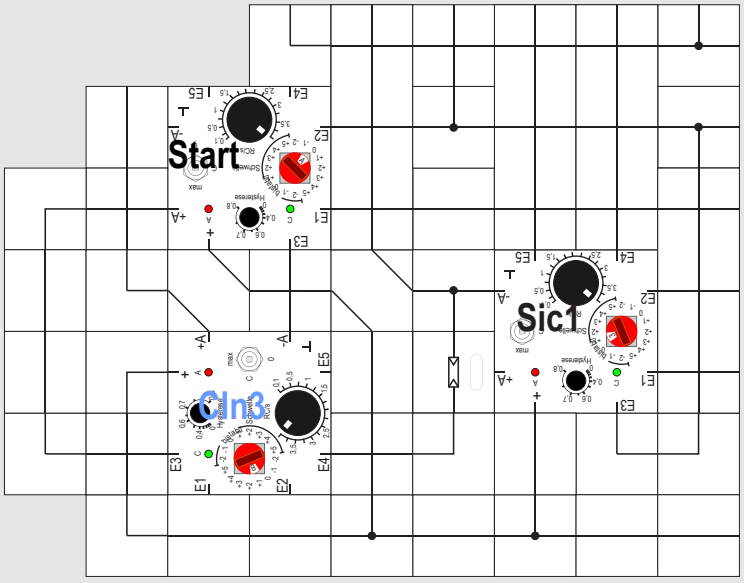

Cln1,2
SBF
MBF
Cdh1
Mcm1
Clb5,6
Clb1,2
Sic1
Cln3
Swi5
Cdc20/14
1,25V
+12V
12V~

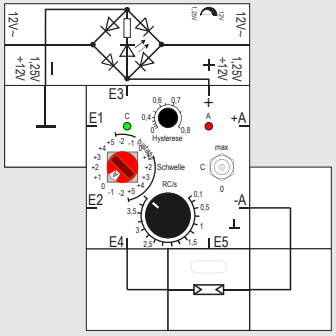
Experiment 28
Light-dependent division of baker's yeast

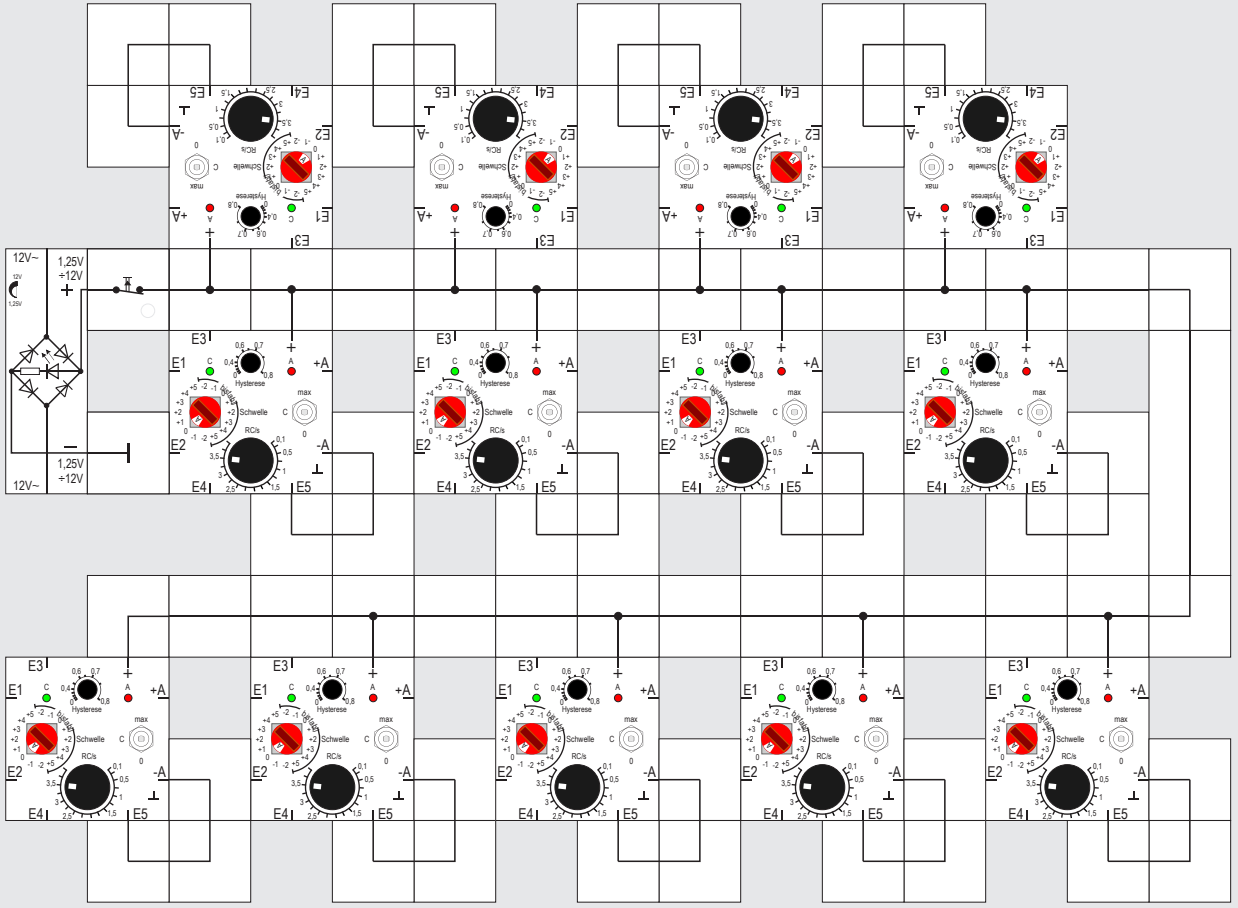
Using the example of yeast we would like to show a simple way to simulate light-dependent processes. By insertion of a photoresistor to the setup, we can ensure that the cyclic restart only occurs with sufficient light. The photoresistor, made of cadmium sulfide, has a resistance of about 100Ω to 500Ω in strong light, and is in fact conductive when connecting an output of one gene module with the input of another. In the dark, the resistance increases to 10MΩ, which acts like an interruption of the line.

If we connect the +A output of **Sic1** via the photo- resistor to an input of **Cln3** (the setup of the yeast network is changed a little to enable this) and increase its threshold to +2, the cycle can only be started if the photoresistor detects enough light.

The picture on the right containing part of the setup shows that the opposite can be done by a little change in the circuit. Now the division will only be started if it is dark. The threshold has to be +1 again. This also works with other circuits like the oscillator (right picture).






12V~
1,25V
÷12V
+
–
1,25V
÷12V
12V~
E1
E2
E3
E4
E5
+A
-A
Hysterese
Schwelle
RC/s
max

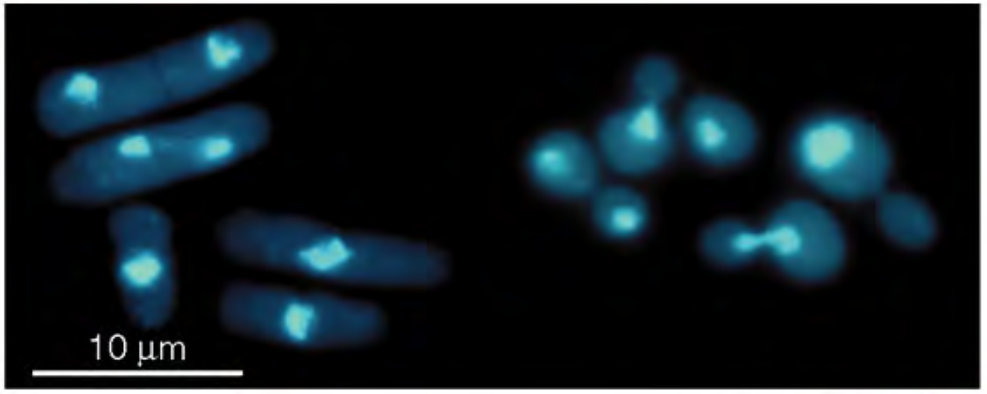
Experiment 29
Fine adjustment of the delay

The big yeast network in our next-to-last experiment has shown us the activation pattern of a real yeast cell, if correctly built, without a lot of meticulous adjustments. That is surprising, isn't it? In the schematic diagram of the simulation model, a standard RC value is given for its base adjustment, which is equal for every gene module. It appeared to work well enough. Actually, this is a basic feature for biological networks. We will observe that they are really modest in their implementation: They don't need vernier adjustments; the opposite is the case. It is sufficient if every gene is approximately simulated. After all, this is simply logical, because in the soup of signals in a cell, the process is even less finicky. Last but not least, this also forms the basis for the secret of this LECTRON construction kit: Our simulations take advantage of this extreme robustness (reliability in the presence of a lot of inaccuracies) of biological circuits.

However, in some experiments it is important that gene modules are equal to each other. For example, if we want to get a precise time history of the previous yeast simulation, or study the dynamics of the networks in slow motion.

The tolerances of the internal capacitors of the gene modules are quite big (20%), which results in a different rate of charging and discharging of the capacitors, although the adjustments are identical. However, we can equalize the modules with a fine adjustment by the following approach: We set the claimed delay and the right hysteresis and start them simultaneously by applying the power supply module. All modules are set as oscillators, so after a short period of time all the red LEDs will flash, but not at the same time because of the tolerances. We choose one module whose LED flashes in a »timely middle« area as a reference and carefully alter the time constants (RC) of the modules that flash sooner or later in the right way. Afterward we remove the power supply module, wait 10 seconds or so (the capacitors have to discharge) and then add it back to our setup. The time differences between the LEDs turning on should now be less, or even zero; they can be further minimized if we allow the modules to oscillate more often, because the tolerances add up and then can better be observed. This way we construct a basic setup with all gene modules identically adjusted.

Fission yeast



Baker's yeast

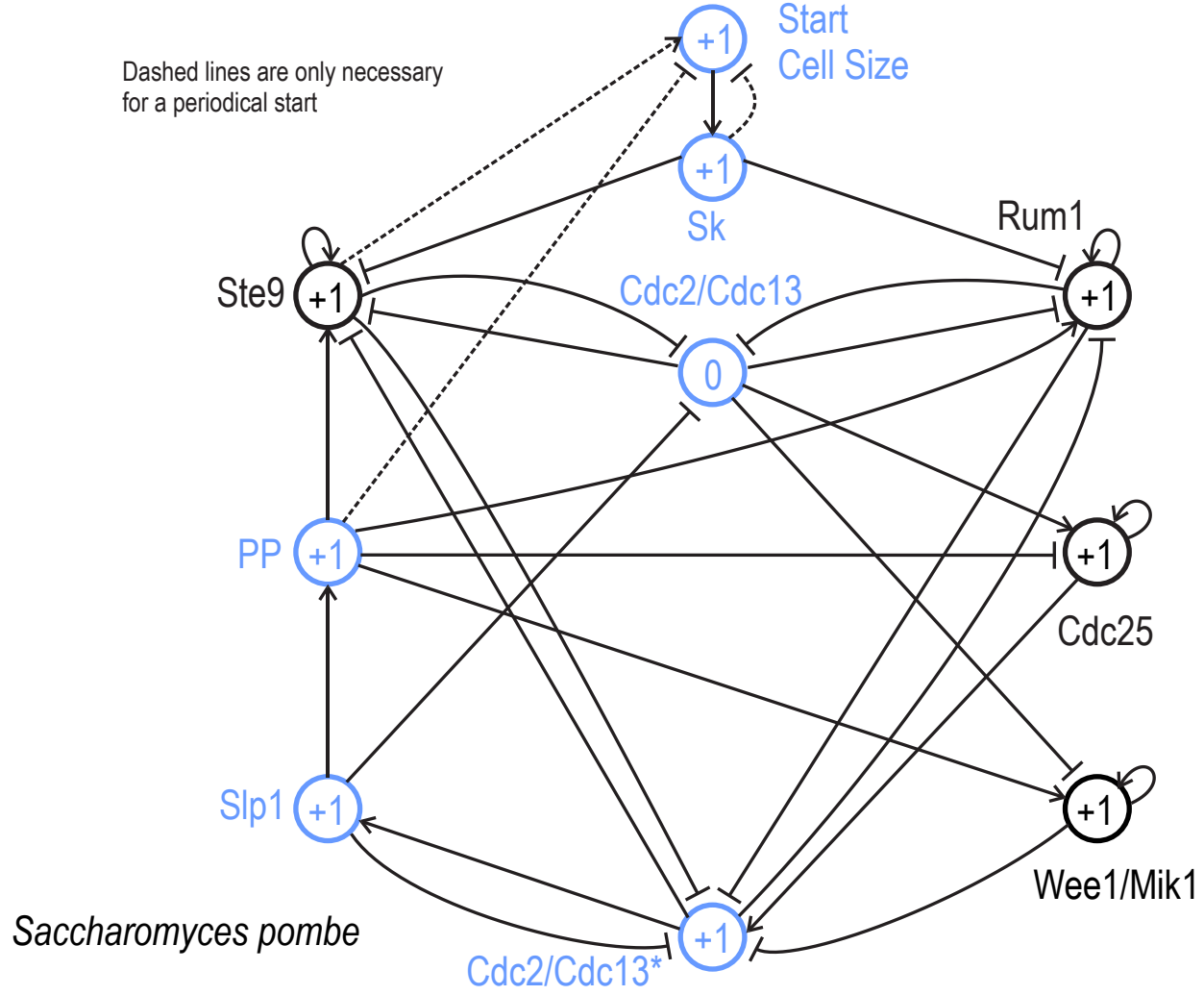
to ↓ from→	Start	SK	RUM1	Wee1	Ste9	Cdc2/13	Cdc25	Cdc2/13*	Slp1	PP	threshold
Start	0	-1	0	0	+1	0	0	0	0	-1	+1
SK	**+1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**+1**
RUM1	**0**	**-1**	**+1**	**0**	**0**	**0**	**0**	**-1**	**0**	**+1**	**+1b**
Wee1	**0**	**0**	**0**	**+1**	**0**	**-1**	**0**	**0**	**0**	**+1**	**+1b**
Ste9	**0**	**-1**	**0**	**0**	**+1**	**0**	**0**	**-1**	**0**	**+1**	**+1b**
Cdc2/13	**0**	**0**	**-1**	**0**	**-1**	**0**	**0**	**0**	**-1**	**0**	**0**
Cdc25	**0**	**0**	**0**	**0**	**0**	**+1**	**+1**	**0**	**0**	**-1**	**+1b**
Cdc2/13*	**0**	**0**	**-1**	**-1**	**-1**	**0**	**+1**	**0**	**-1**	**0**	**+1**
Slp1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**+1**	**0**	**0**	**+1**
PP	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**+1**	**0**	**+1**

Experiment 30
Simulation model of fission yeast

The fission yeast *Schizosaccharomyces pombe*, whose propagation cycle we want to simulate next, is another type of fungus that reproduces itself, but in this case, not by budding of daughter cells, but by division of the cell into two halves (see figure) [13]. It is an elongated single-celled eukaryote that is also used as a model organism in molecular and cell biology. The genomic DNA sequence of this yeast was published in 2002.

The table shows that its division can be simulated with nine gene modules. In addition to that, we will again use a Start module to make the network run periodically. Just as for the first yeast, the weights of the signals are only +1 or -1. The thresholds are similar, too.

To make the setup with the gene and connection modules easier, we have transformed the matrix in the table to a graphic. It is displayed on the next page.

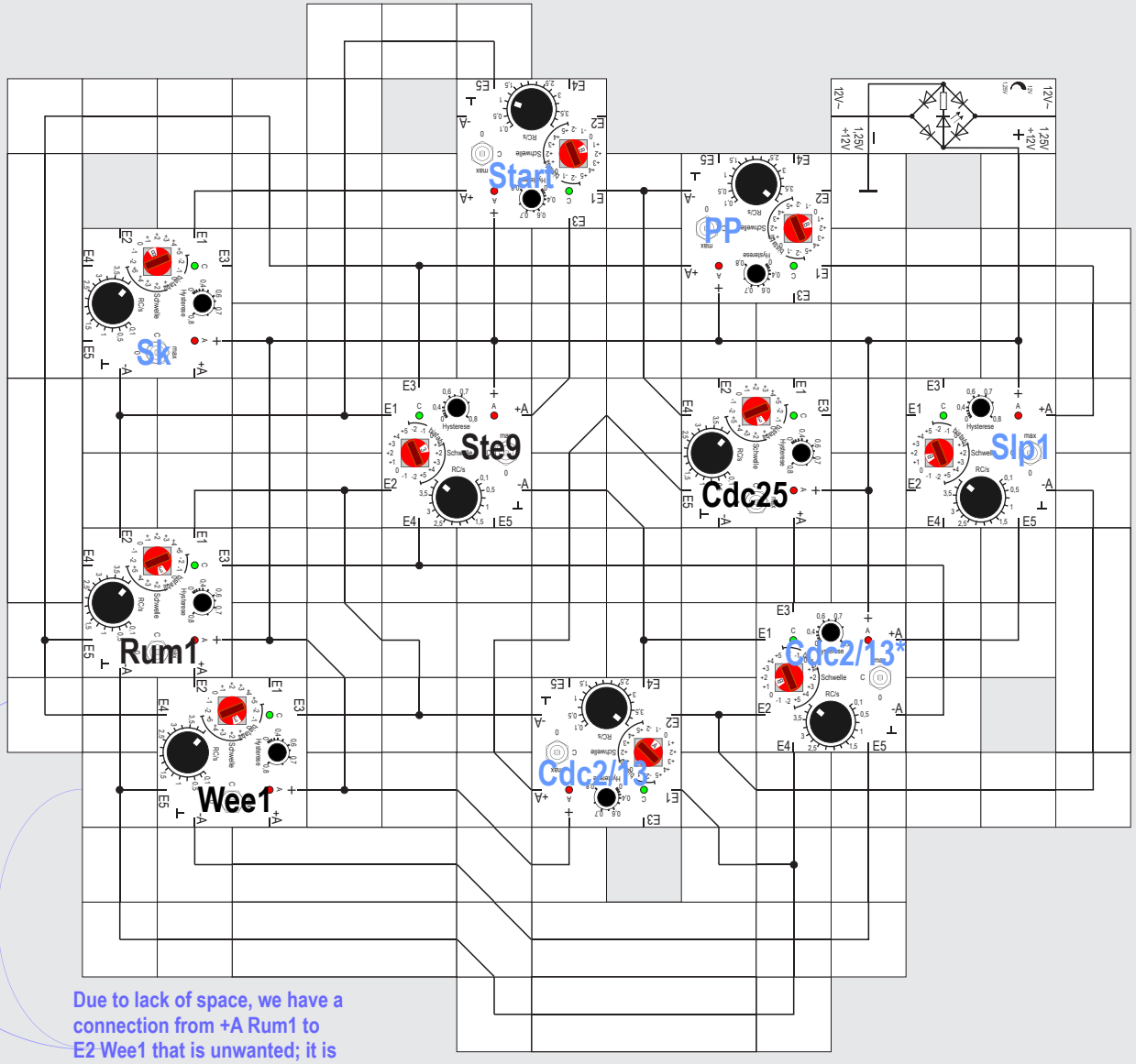


Saccharomyces pombe

One possible setup for the simulation resulting from the connection diagram is displayed on the next page. Since we need one gene module less, the setup is a little easier than it was for the first yeast. There is a specific feature noted in the setup diagram. Aside from that, all controls and precautions about setting up the operation of the first yeast cycle simulation apply.
If the simulation runs without errors, the sequence of activity and inactivity in the table should occur. The time diagram shows a possible run that is already optimized concerning small transition areas. The period is 87 seconds.
Here we can also be quite sure that changing thresholds or permanently activating or inactivating gene modules have a biological counterpart that leads to viable or nonviable mutations.

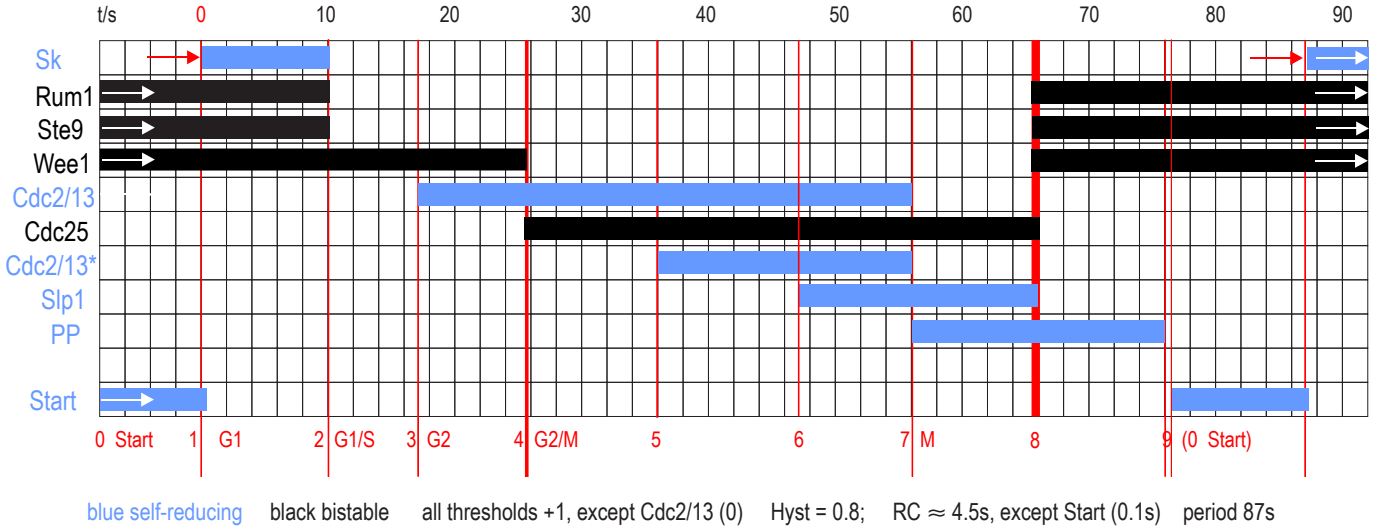
Mutations

We take the fission yeast as an example to take a look at how we can simulate mutations with gene modules [14]. Thus we deactivate single genes in the correctly working model (which represents the so-called wild type of fission yeast) by switching them to the »0« position and restarting the cycle. The most exciting question is whether a mutant will



Due to lack of space, we have a connection from +A Rum1 to E2 Wee1 that is unwanted; it is compensated for by the connection from -A Rum1 to E5 Wee1.

Step	Start	Sk	Rum1	Ste9	Wee1	Cdc2/13	Cdc25	Cdc2/13*	Slp1	PP	Phase
0	1	0	1	1	1	0	0	0	0	0	Start
1	0	1	1	1	1	0	0	0	0	0	G1
2	0	0	0	0	1	0	0	0	0	0	G1/S
3	0	0	0	0	1	1	0	0	0	0	G2
4	0	0	0	0	0	1	1	0	0	0	G2
5	0	0	0	0	0	1	1	1	0	0	G2/M
6	0	0	0	0	0	1	1	1	1	0	G2/M
7	0	0	0	0	0	0	1	0	1	1	M
8	0	0	1	1	1	0	0	0	0	1	M
9	0	0	1	1	1	0	0	0	0	0	G1

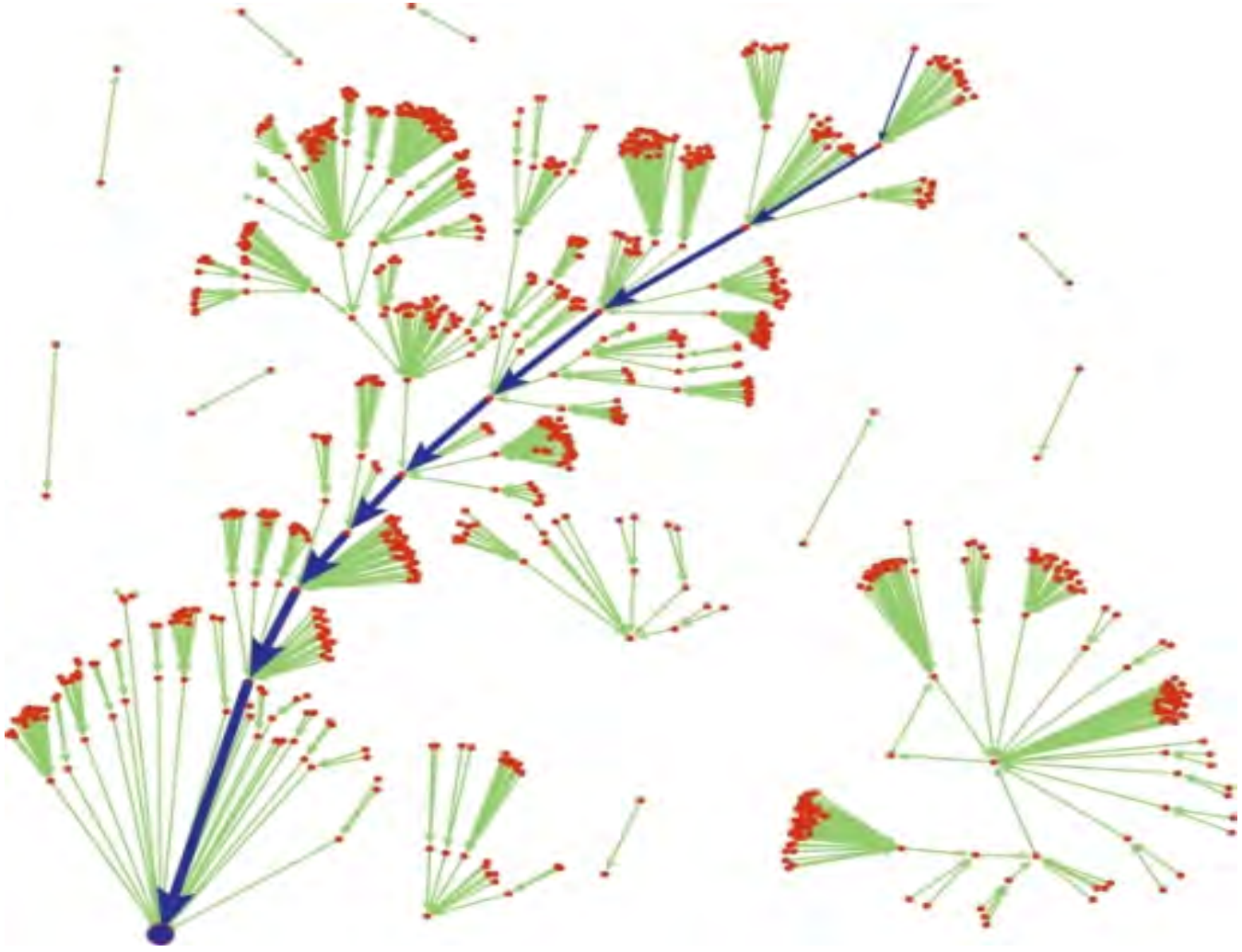


be viable and, in particular, capable of replication or if the cell cycle gets stuck in an intermediate state (or fixed point).

Let's take a close look at some special mutants. If you deactivate **Wee1**, the cell cycle seems to run without problems. You can see the same things happening as without the mutation, so this system is viable. Real mutants of fission yeast that are created by a knockout of the **Wee1** gene act as predicted: they divide and are viable!

If we deactivate the three genes **Ste9**, **Rum1**, and **Wee1** at once, an oscillation between the other genes (except for **SK** and **Start**) occurs. This is equivalent to a permanent M phase. The cell should divide without going through the other phases in which the cell grows and replicates its DNA. The mutation forms diseased tissue and therefore is not viable. In fact, real fission yeast with this triple knockout won't divide.

If **Wee1** and **Cdc25** are deactivated in a double knockout, there will soon be a fixed point where only **Cdc2/13** is active. Because **Cdc2/13*** is not activated, the M phase cannot start, so the system is stuck in G2 phase. This means it keeps growing without dividing. This mutant isn't viable, either. The same happens when only **Cdc25** is deactivated.



This indicates that **Cdc25** is the most important gene to enter the M phase. This is consistent with the interaction matrix because **Cdc25** activates gene **Cdc2/13***, which is important for the M phase. If you deactivate **Cdc2/13**, there will also soon be a fixed point with only **Wee1** activated, because **Cdc2/13** will activate **Cdc25**, so the M phase can start. Since this does not happen, the cell is not viable.

If you now deactivate **Ste9**, the cell cycle runs once and then remains in a fixed point with only **Rum1** and **Wee1** activated. Because it is **Ste9** that reactivates **Start** in our model, the cell cycle can't be restarted without help. So the model remains in G1 phase. But the mutation is viable because it runs through the whole cycle, and in real cells, the start signal is made by the growth of the cell.

Experiments show that the real **Ste9** knockout mutant is capable of replication. If only **Rum1** is deactivated, an oscillation between all genes except for **Start**, **Ste9**, and **SK** occurs.

Theoretically a cell cycle with only **Rum1** deactivated should be no problem, and indeed this is what the real **Rum1** mutant does. Finally, we deactivate **SIP**. A fixed point occurs with **Cdc2/13**, **Cdc2/13***, and **Cdc25** active. **SIP** is responsible for activating **PP**, which activates **Wee1**, **Rum1**, and **Ste9**, so it restores the steady state. Without this gene the G1 phase cannot start, and the system is not viable. This has been observed in nature, too.

These mutations show that interference with certain parts of the network has a more or less heavy impact on the cell cycle, and we can comprehend this with our model. We are even able to predict if a certain mutant will be viable or not. It surely is amazing what this little experimental kit can do, isn't it?

Attractor of fission yeast

Just like baker's yeast, there is a graphically impressive attractor. It is displayed above and contains, taking into account the Start gene, $2^{10} = 1024$ states. This time all of them are displayed. The usual path is marked clearly with blue arrows; those are the states we find in the table. The attractor is not quite as dominant as in baker's yeast. Still, 73% of the possible states lead to the same final state of the cell cycle.

Experiment 31
Simulation of Arabidopsis

Arabidopsis thaliana (thale or wall cress) is a plant in the cabbage family. According to an online encyclopaedia, »Arabidopsis is native to Europe, Asia, and northwestern Africa. It also appears to be native in tropical afroalpine ecosystems. It is an annual (rarely biennial) plant, usually growing to 20-25 cm tall. The leaves form a rosette at the base of the plant with a few leaves also on the flowering stem. The flowers are 3 mm in diameter, arranged in a corymb [a cluster with all the flowers at the same level]; their structure is that of the typical Brassicaceae [cabbage family].«

We want to deal with the plant and simulate the development of its blossoms. Why this plant in particular? Well, in the 1940s it was chosen as a genetic model organism and by now it is the »house plant« of molecular geneticists. With its short life cycle, it is well suited for experiments on mutations. From our perspective, no other plant is studied more than *A. thaliana*.

The development of the different petals is regulated by the interaction of a few genes. To simulate this, we want to use the ABC model. This model contains three groups: A, B and C. For the sake of convenience, the A group just contains the gene **AP1**. Normally the gene **AP2** is also included, but it is not as important, so we will ignore it.

Group B contains the genes **AP3** and **PI**. The last group, C, only contains **AG**. A and C inhibit each other; therefore, if C is active, A is not and vice versa. The genes in A and both of the genes in group B control how the blossoms develop. To simulate this, we choose a model that was published in 1998 by Luis Mendoza and Elena R. Alvarez-Buylla [15].

1. When the A genes are activated, the sepals grow.
2. When the B genes are also activated later on, the combination of both groups ensures the development of the blossoms.
3. If C is active, A is inhibited. At that point, B and C are able to develop the stamens.
4. When B becomes inactive, C is responsible for developing the carpels.

The published model which we want to use for our simulation has to be adapted to the circumstances because some attributes will not be implemented off-hand. The gene modules just have five inputs; however, **LFY** and **AG** need six input signals, **PI** and **AP3** need seven, and **AP3** needs eight. The corresponding matrix of all genes and their interconnections, which still needs to be adapted, is shown in the usual way.

With some consideration it is possible to change the matrix in a way that our gene modules can handle the situation.

1. To turn off **TFL1**, a -2 signal is not needed; -1 is enough. **TFL1** is activated when the gene **EMF1** is activated. **TFL1** is not activated when **LFY** gives only a -1 signal (instead of -2). This process doesn't depend on **EMF1**! Therefore, we save one input.
2. The gene **LFY** needs the signal +4; however, its maximal input is +3 (+2 from **AP1** and +1 **CAL**). The missing signal has to be generated manually by switching the toggle switch to »max«. The **LFY** module turns off shortly after switching back to »C«.
3. The threshold of **AP1** is 0. The gene modules **EMF1** and **AG** each deliver a -1 signal (this adds up to -2). To turn **AP1** on, the +2 signal from **LFY** is enough, which normally provides a +5 signal.
4. **CAL** just gets a +1 signal from **LFY**, instead of +2. We decrease the threshold from +2 to +1, so it fits again.
5. **AG** actually gets six input signals. That is one too many. **TFL1** and **AP1** separately turn off **AG** with their -2 signals. In an OR operation, an additional gene module (**OR**) generates a single signal from these two signals that sends the -2 signal to **AG** if at least one of its input signals (**AP1**, **TFL1** or both) = 1.
6. Eventually it is possible to decrease the weights of the signals that activate **AP3** and **PI** without

changing anything essential. These two signals are weighted +3 and +4, respectively, from **LFY**. They are now decreased to +2.
In fact, **UFO** gives a +2 signal to **AP3**; however, a +1 signal is sufficient, and finally the -2 output signal from **SUP** to **AP3** is changed from -2 to -1.
Unfortunately we had to use a gene module for the OR connection; therefore, one module is missing for the setup. We need 13 modules, but some genes do not influence the dynamic as strongly as others, so therefore we can omit them. These are **LFY** and **UFO**. We have decided to omit **UFO**.
We will mention that it is possible, too, to build the OR operation with two Schottky diodes instead of a gene module. In that case, this module would be free to use. We will get back to this special setup later (without using **UFO**), though. That circuit should only be built by people with basic knowledge of electronic engineering. A reverse connection of the diodes can lead to destruction of the circuit, because then both outputs work against each other.
The modified matrix for simulating the blossom development of *A. thaliana* then looks as shown, with the modifications marked in red. The connection diagram is illustrated on the next page to help manage the setup without errors. The illustration of a possible compact setup then follows.

to ↓ from→	EMF1	TFL1	LFY	AP1	CAL	LUG	UFO	BFU	AG	AP3	PI	SUP	Threshold
EFM1	0	0	0	0	0	0	0	0	0	0	0	0	+1
TFL1	+1	0	-2	0	0	0	0	0	0	0	0	0	+1
LFY	-2	-1	0	+2	+1	0	0	0	0	0	0	0	+4
AP1	-1	0	+5	0	0	0	0	0	-1	0	0	0	0
CAL	0	0	+2	0	0	0	0	0	0	0	0	0	+2
LUG	0	0	0	0	0	0	0	0	0	0	0	0	+1
UFO	0	0	0	0	0	0	0	0	0	0	0	0	+1
BFU	0	0	0	0	0	0	0	0	0	+1	+1	0	+2
AG	0	-2	+1	-2	0	-1	0	0	0	0	0	0	0
AP3	0	0	+3	0	0	0	+2	+1	0	0	0	-2	+1
PI	0	0	+4	0	0	0	+1	+1	0	0	0	-1	+1
SUP	0	0	0	0	0	0	0	0	0	0	0	0	+1

Note: The construction of the gene module allows us to connect unused inputs of different modules (in contrast to their outputs) without affecting the correct function of what is being set up. That is why both modules **CAL** and **SUB** can be closely spaced.

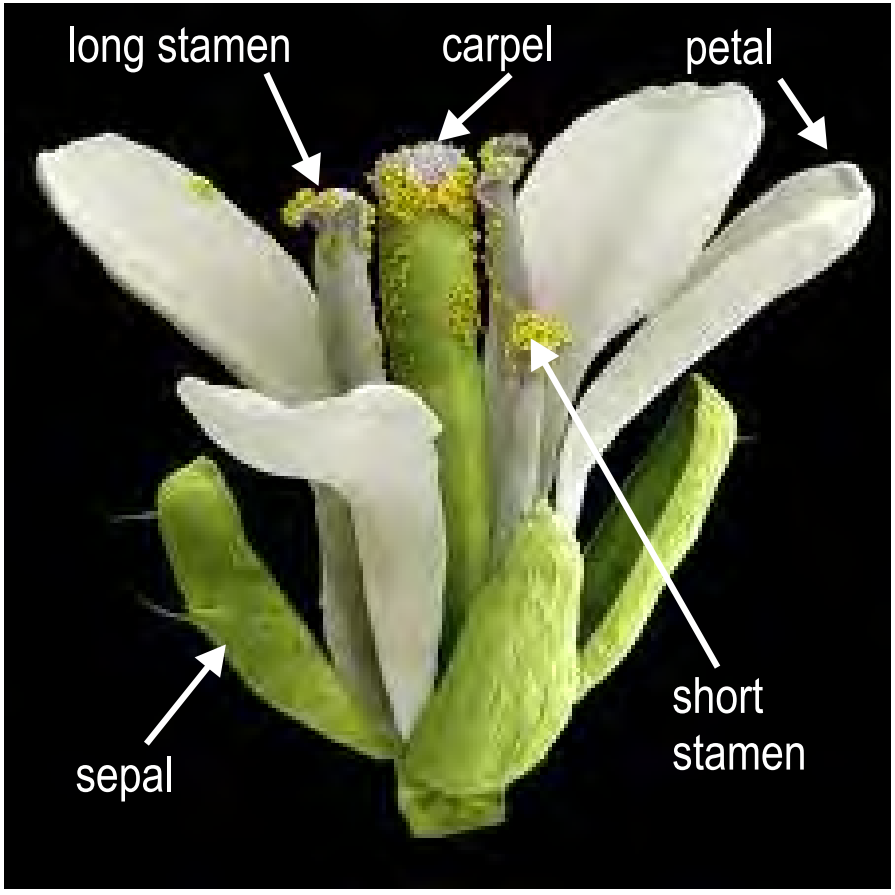
The simulation of flower development, or morphogenesis, consists of six more-or-less independent steps that have to occur individually. These single steps are:

1. Activation of gene A (**AP1**).
2. Activation of genes B (**AP3** and **PI**).
3. Inhibition of gene A (**AP1**).
4. Activation of gene C (**AG**).
5. Inhibition of genes B (**AP3** and **PI**).
6. Inhibition of gene C (**AG**).

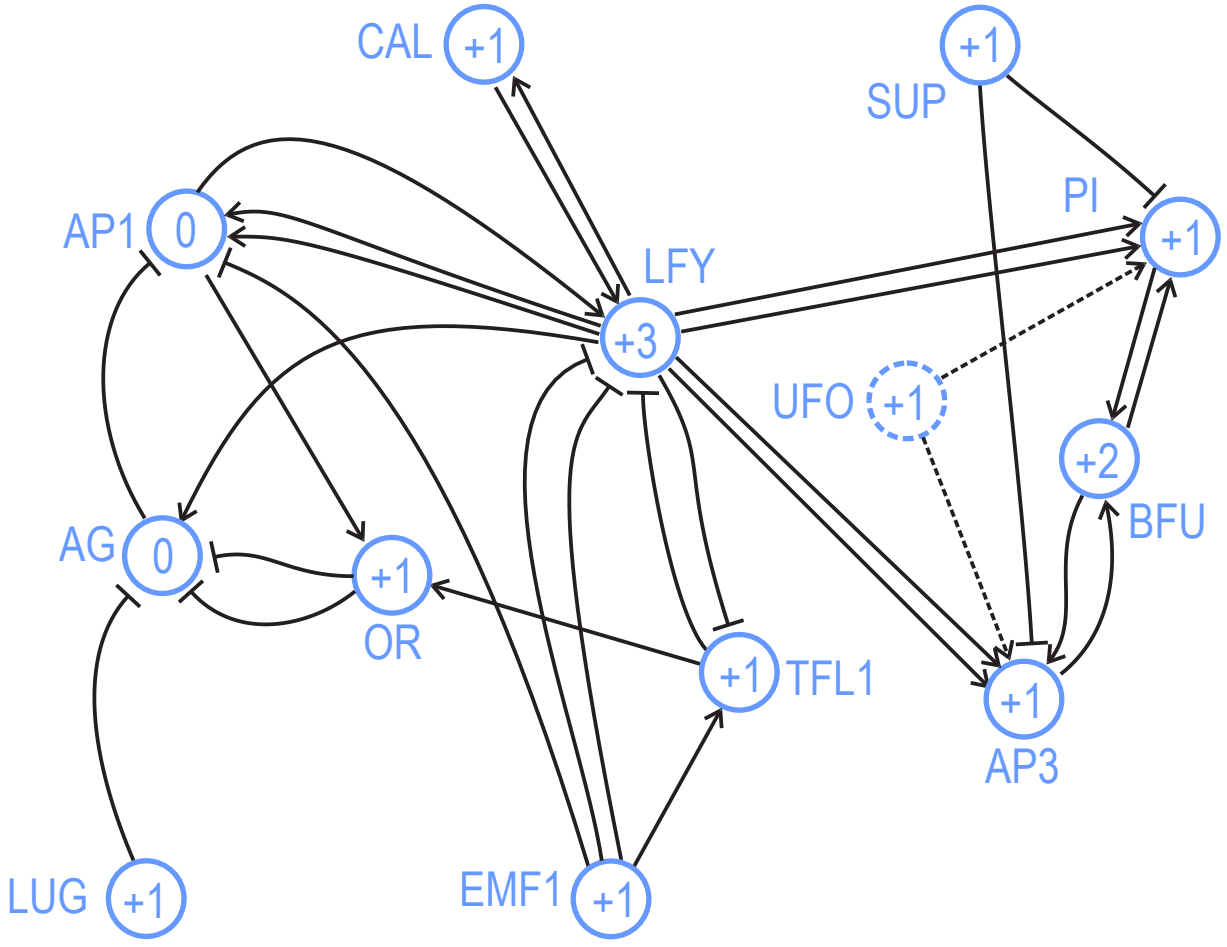
The following shows how to handle the single steps in morphogenesis.

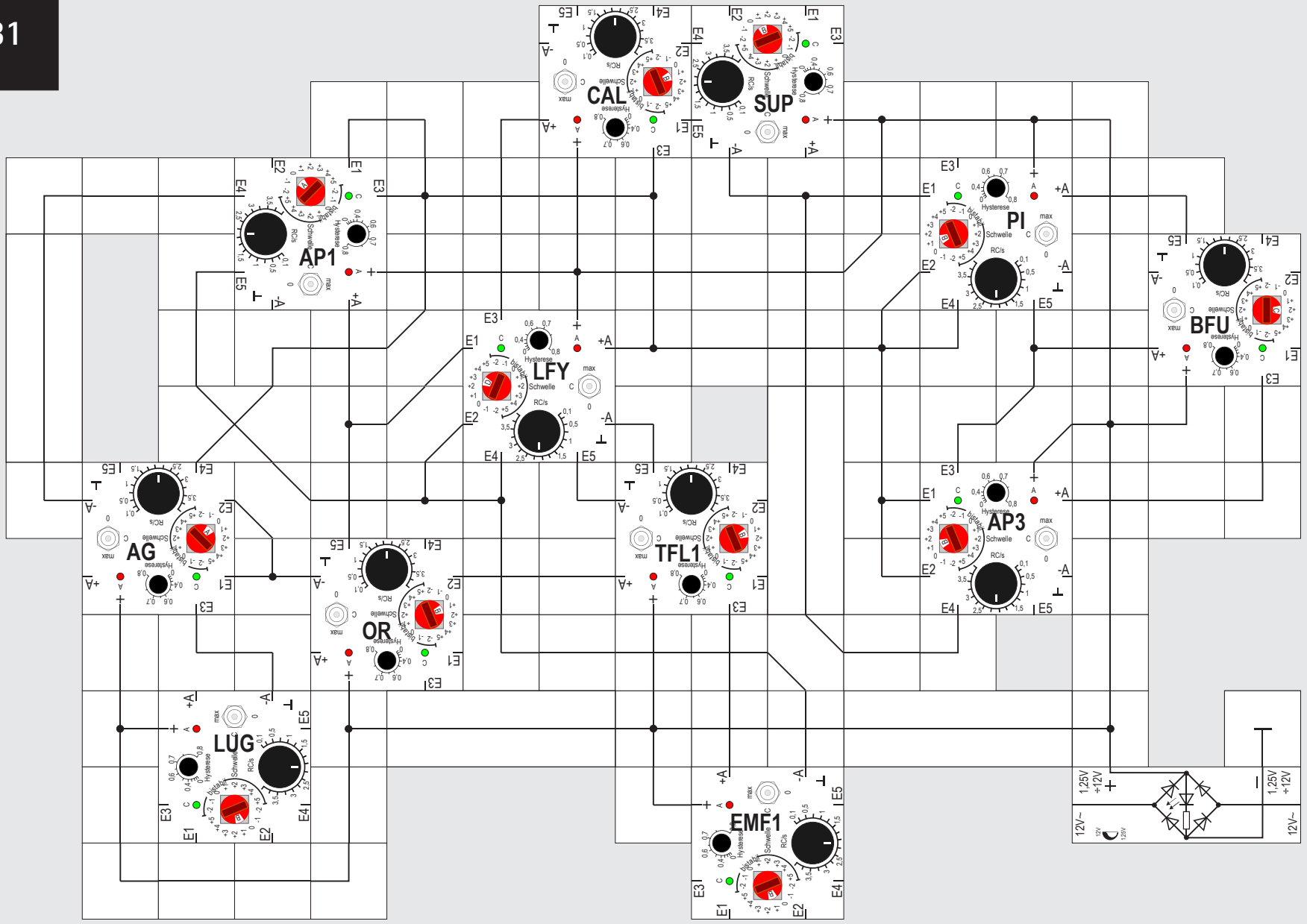
At the beginning of flower morphogenesis, gene **EMF1** is activated by adjusting the toggle switch to »max«. Thus gene **AP1** in A is inhibited. But **EMF1** also activates **TFL1**, which again activates the »**OR** gene« and thereby inhibits **AG** in C.

to ↓ from→	EMF1	TFL1	LFY	AP1	CAL	LUG	UFO	BFU	AG	AP3	PI	SUP	Threshold
EFM1	0	0	0	0	0	0	0	0	0	0	0	0	+1
TFL1	+1	0	−1	0	0	0	0	0	0	0	0	0	+1
LFY	−2	−1	0	+1	+1	0	0	0	0	0	0	0	+3
AP1	−1	0	+2	0	0	0	0	0	−1	0	0	0	0
CAL	0	0	+1	0	0	0	0	0	0	0	0	0	+1
LUG	0	0	0	0	0	0	0	0	0	0	0	0	+1
UFO	0	0	0	0	0	0	0	0	0	0	0	0	+1
BFU	0	0	0	0	0	0	0	0	0	+1	+1	0	+2
AG	0	−2	+1	−2	0	−1	0	0	0	0	0	0	0
AP3	0	0	+2	0	0	0	+1	+1	0	0	0	−1	+1
PI	0	0	+2	0	0	0	+1	+1	0	0	0	−1	+1
SUP	0	0	0	0	0	0	0	0	0	0	0	0	+1



Arabidopsis thaliana




CAL
SUP
AP1
PI
LFY
BFU
AG
TFL1
AP3
OR
LÜG
EMF1
1,25V
+12V
12V~

Activation of gene AP1 in A

First the toggle switch of gene-modul **LUG** has to be switched to **»max«**, so gene **AG** in C gets inhibited. Otherwise, C would inhibit A constantly. To activate gene **AP1 EMF1** has to be deactivated manually. This also deactivates **TFL1** and **OR**. Gene **AP1** in A starts activating due to **EMF1** being turned off. Gene A enables **OR** which inhibits C. This happens quite slowly because the RC-time is set to > 1.5s. Later **LUG** can be set arbitrarily because **OR** will inhibit C continuously. The table represents the different states of all genes; 1 means activated gene, and 0 means that the gene is either deactivated or inhibited. At the end, gene A, **(AP1)** remains activated, and the development of the sepals occurs.

t	EMF1	TFL1	AP1	LUG	AG	OR
0	1	1	0	1	0	1
1	0	1	0	1	0	1
2	0	0	0	1	0	1
3	0	0	0	1	0	0
4	0	0	1	1	0	0
5	0	0	1	-	0	1

Activation of genes in group B

The next table shows in the same manner the development of the dynamics with respect to the B genes.

t	LFY	PI	AP3	BFU	
0	1	0	0	0	LFY1 has to be activated manually. AP3 and PI are still turned off.
1	1	1	1	0	LFY1 activates both B genes AP3 and PI
2	1	1	1	1	These two offer BFU the incoming signal to become activated
3	0	1	1	1	LFY1 can get switched off while BFU keeps the two B genes active.

To this point, both B genes are activated and build petals together with A.

Inhibition of genes in group A

When A gets deactivated it does not inhibit gene C (**AG**) anymore.
The table shows the sequence. In this case it is quite easy to inhibit A

t	EMF1	TFL1	OR	AP1	
0	0	0	1	1	This is still the initial state: **AP1** and **OR** are still on; **EMF1** is still off
1	1	0	1	1	**EMF1** gets activated manually
2	1	1	1	1	**TFL1** is activated by **EMF1**
3	1	1	1	0	**TFL1** activates **OR** and inhibits gene A directly

Activation of gene C

Now A is deactivated. Usually C would not be inhibited anymore. But in this case **EMF1** has to be turned off again. Then **TFL1** and later **OR** get deactivated as well. Although A is not inhibited anymore, it cannot activate **OR** fast enough, so this building block does not inhibit C. This happens because the RC time of A is higher than the RC times of other blocks. C gets activated due to its threshold of 0 and the missing signal from **LUG**. When C is active A has no chance to remain on and provide an inhibiting signal to OR. C remains active.

t	EMF1	TFL1	OR	AP1	AG	
0	0	1	1	0	0	**EMF1** gets deactivated once
1	0	0	1	0	0	Thus **TFL1** turns on as well
2	0	0	0	0	0	As a result, there is no incoming signal for **OR**
3	0	0	0	0	1	At the end C activates itself while inhibiting A

Together, gene C and genes B build stamens.

Inhibition of the B genes

To inhibit the B genes, it is necessary to activate **SUP** manually. This inhibits both B genes, **PI** and **AP3**.
These genes cannot offer **BFU** the needed signals, and therefore **BFU** cannot keep both B genes active.
At the end, **SUP** can be set arbitrarily.

t	SUP	PI	AP3	BFU	
0	0	1	1	1	First the two genes **AP3** and **PI** (group B)and **BFU** are still active
1	1	1	1	1	SUP has to be activated manually
2	1	0	0	1	The signal will inhibit both B genes, **AP3** and **PI**
3	1	0	0	0	**BFU** does not get signals anymore and becomes deactivated
4	-	0	0	0	**SUP** can be turned off again; the B genes will not get activated anymore

Due to missing B, gene C starts to build carpels.

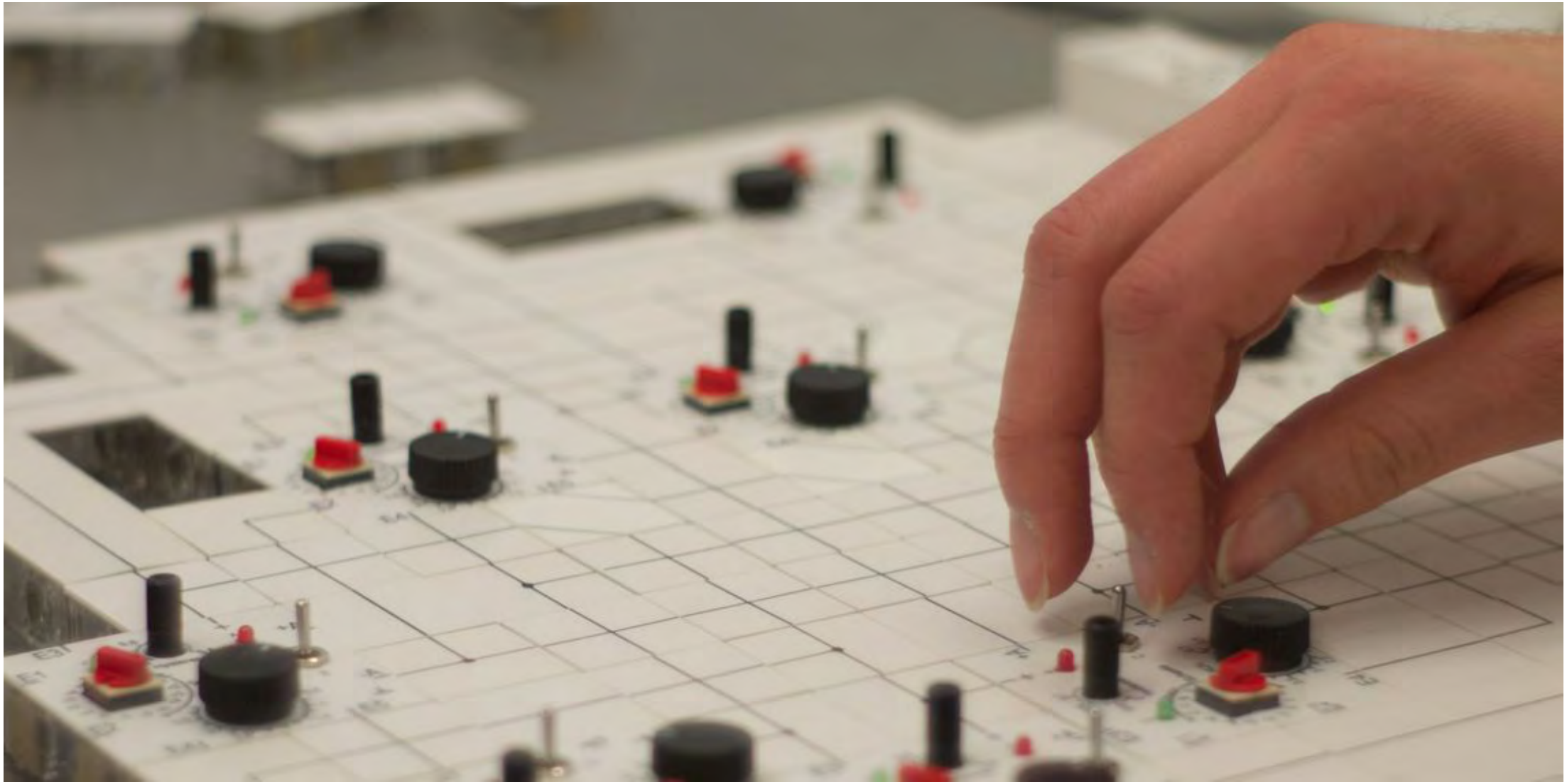
Inhibition of gene C

To complete flower morphogenesis, C has to be enabled again. Afterward, the sequence can repeat itself.
Manual activation of **EMF1** leads to the activation of **TFL1** and **OR**. The last electronic gene building block, **OR,** inhibits C. In the end, all genes that affect the formation of flowers (flower morphogenesis) are enabled.

t	EMF1	TFL1	OR	AG	
0	0	0	0	1	**AG** (group C) is the only active gene.
1	1	0	0	1	**EMF1** is activated manually.
2	1	1	0	1	**TFL1** is activated by **EMF1**.
3	1	1	1	1	**TFL1** activates **OR**.
4	1	1	1	0	**OR** turns off **AG** (group C).

Unfortunately the simulation of flower morphogenesis is not self-releasing due to the missing inputs of the blocks.

Lectron

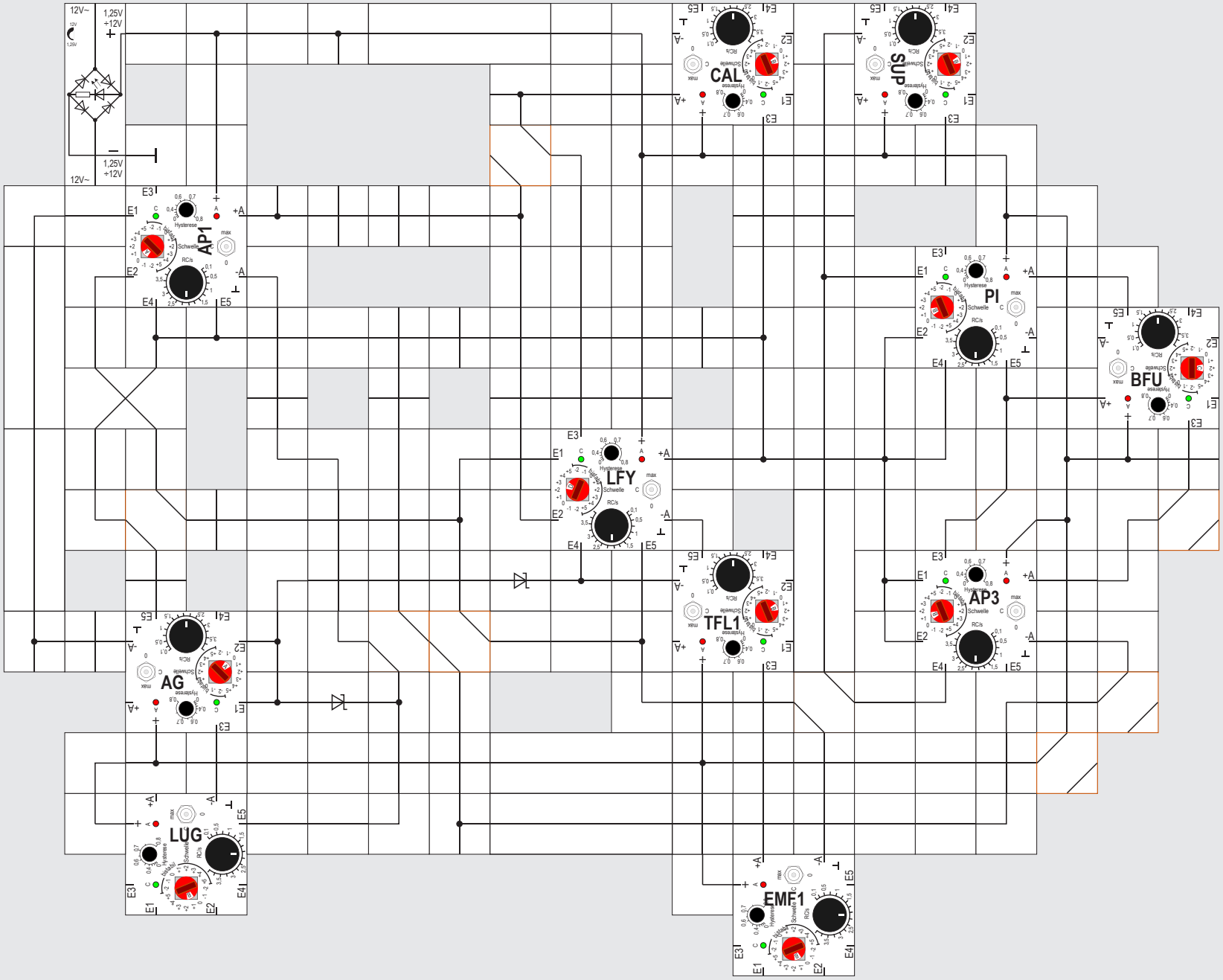


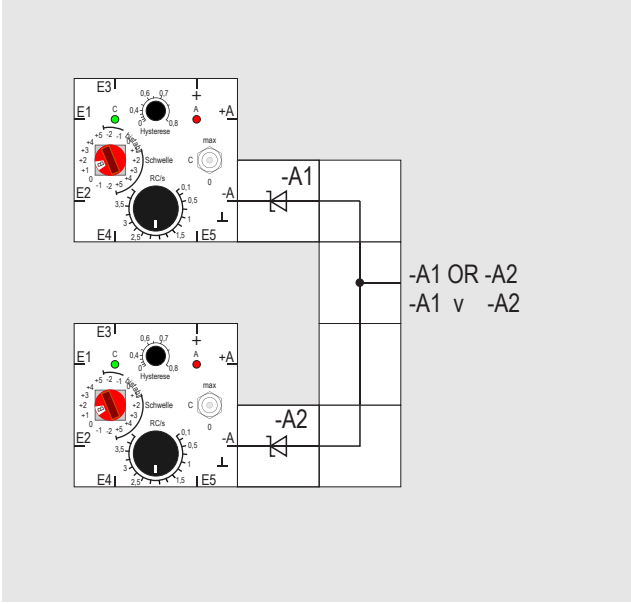
Digression
Automatic processing of the sequence of flowering

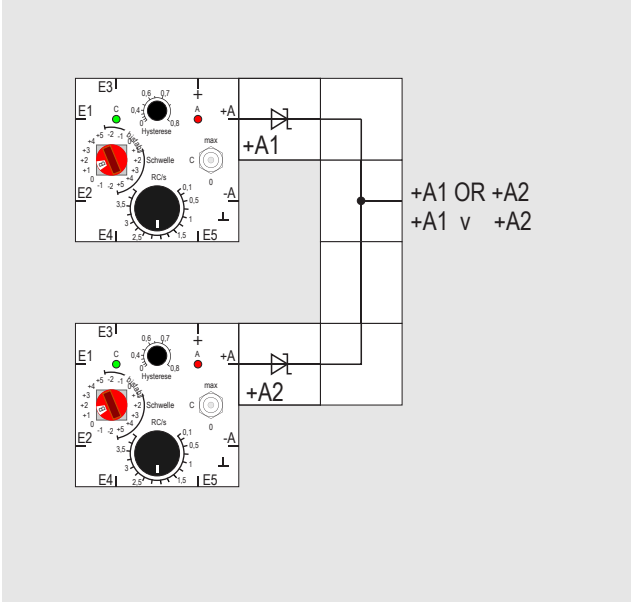
Of course it is not very elegant to use manual switches to continue a simulation. During the blossom's emergence in the real flower, external signals ensure that the next step starts when the previous one was properly executed. We have to simulate these signals in our model manually.
Another possibility would be to generate these external signals artificially each time from the previous step, as we did in the yeast network with the Start module when a complete round of cell growth started the next cycle. In a little digression, we will implement such a procedure with the following experiments.
Those who are more interested in modeling other features in biology can skip this part and continue with the next biological example, the larva of the flour beetle in experiment 39.
Those who want to follow this digression and learn something about the dynamics of autonomous networks will be rewarded with an aesthetically pleasing blinking Arabidopsis network. We will need additional logical connections, which we will simulate in the manner of electrical engineering by diodes and resistors.
To do this, we need to rebuild the setup (see next page). It looks full of holes, but we will fill the holes soon.

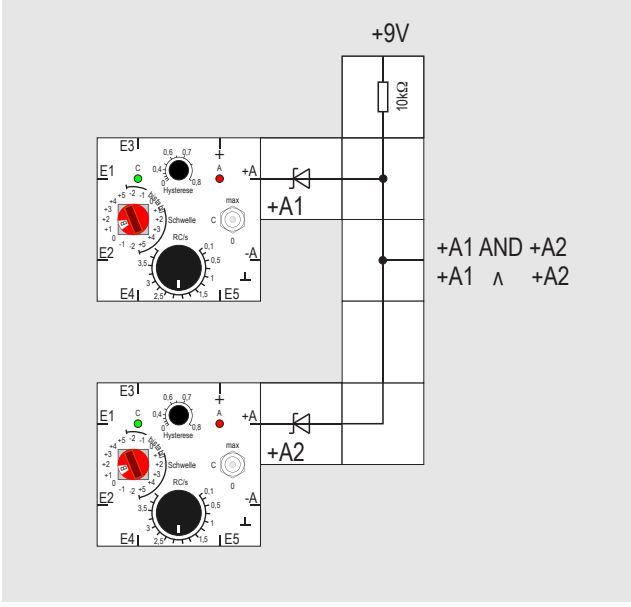
Experiment 32
Signal combination by diodes

First we replace the gene module **OR**, which simply combines two signals, with a diode OR logic element that is often used in digital technology. We get the circuit shown and free up a gene module for further application. Schottky diodes qualify best for this purpose due to their low threshold voltage (U_{th} ≈ ?0.2V). When we test the modified circuit, we will see that it processes similarly to the original one.
Now on to how the OR operation works. When at least one of the -A signals of **AP1** OR **TFL1** is active, e.g., when one signal output has -8V, we want **AG** to get an inhibiting signal with weight -2. Connecting the diodes as shown, a negative output potential at one of the involved gene modules dictates the potential of the diode **OR** logic element. The possible 0V potential of the other gene module doesn't interfere: The corresponding diode blocks and decouples the outputs. The resulting signal has to be connected to two inputs of the **AG** module because the weight is -2. This circuit can be extended to more signals if a receiving module doesn't have enough inputs. But before that we have to take a closer look: It is ideal if the signals that have


12V~
1,25V
÷12V
12V~
1,25V
÷12V
AP1
CAL
SUP
PI
BFU
LFY
TFL1
AP3
AG
LUG
EMF1

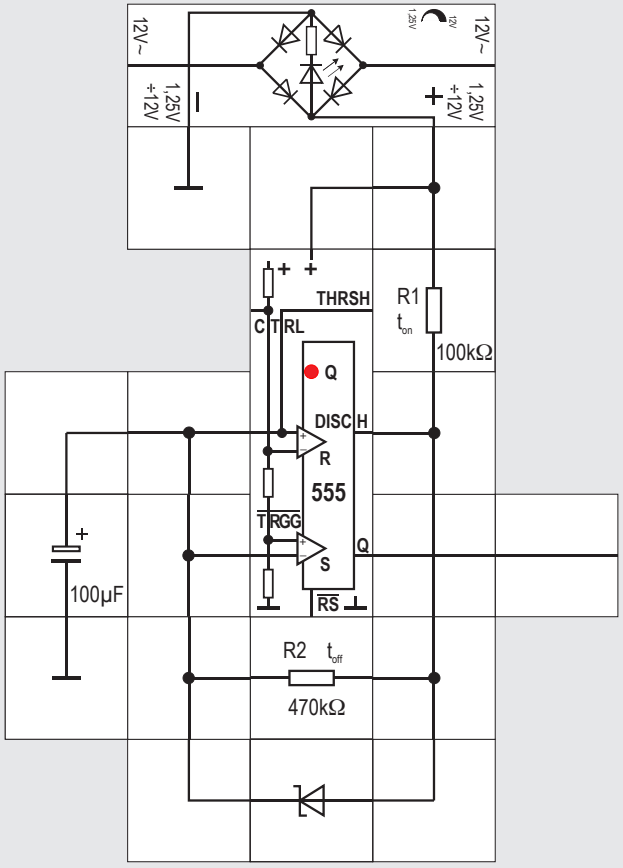






to be combined are active at different times. Then they use one input together. Being active at the same time gives them a lower weight, and we will have to check whether it's possible to compensate for this by changing the threshold.
If you want to combine two or more +A signals with an OR conjunction, the diodes have to be rotated 180°. Then the high potential always determines the result without being influenced by a possible 0V output signal at the other module (see the middle figure).

It becomes a little more complex if we want to combine two or more +A signals in an AND operation. A resistor and one diode per signal are required. The circuit works like this: Only if all output signals are active (8V) is the combined signal 8V, too. As long as there is at least one 0V output, this one determines the combined signal.
If all involved signals are active (8V), all diodes are blocked; the connected module now gets its high potential from the resistor, which therefore may not be too large since its value will be added to the 100kΩ input resistance. However, it may not be too small either, since otherwise the cross-current that a 0V output has to cope with becomes too strong. Suitable values are 5.6kΩ or 10kΩ.
The AND operation of -A signals again requires 180° rotated diodes and a resistor, this time connected to a -8V potential, which usually isn't available and would have to be produced by an additional electronic circuit.


12V~
1,25V
+12V
12V~
1,25V
+12V
THRSH
CTRL
R1
t_{on}
100kΩ
Q
DISC
H
R
555
TRGG
Q
S
RS
100µF
R2
t_{off}
470kΩ

Experiments 33 & 34
Clock generator with timer

When simulating the development of different flower leaves of *A. thaliana*, we notice quickly that the switch at the **EMF1** gene module is often used manually and almost always when a new round of development of the four floral organs should begin. This is what we want to change and therefore automate.

The leftover OR gene module can be used as an oscillator, as we know from before. It would generate a signal that has a duty cycle $t_{on}/T = 0.5$, dependent on its RC adjustments, but that would not be optimal. It would be better to have a signal that is »on« for a short period of time and »off« for a long time, e.g., with a smaller duty cycle of 0.1 or even less.

For this purpose, the LECTRON timer module, which contains the well-known integrated circuit 555, is ideally suited.

In the setup with this module, the values of the capacitor and the resistor for the time determining RC part of the circuit are chosen as C = 100µF and R1 = R2 = 10kΩ; so after connecting the power supply, the operation of the circuit can be observed from the blinking of the LED. The output Q gives a square-wave signal with the desired properties.

If you are interested in electronic engineering, briefly, this is how this is achieved: At the starting point, the capacitor is empty and therefore the signal $\overline{TRGG} = U_L$. The »lower« operational amplifier, working as a comparator, therefore generates a set signal for the flip-flop; Q gives the U_H potential and the LED flashes. DISCH has high resistance. The capacitor charges slowly by an exponential function via R1 and the Schottky diode, which bypasses R2.

When the voltage is 2/3 of the power supply voltage, the »upper« comparator generates a reset signal for the flip-flop. Therefore, $Q = U_L$, the LED turns off and an internal MOSFET conducts current. Consequently DISCH has almost the ground potential, and the capacitor discharges via R2.

When the voltage of the capacitor is less than 1/3 of the supply voltage, the »lower« comparator becomes active and its signal turns on the output Q to the U_H potential; the LED flashes, DISCH becomes high resistance and everything starts again.

With the exception of the starting after connecting the power supply, the capacitor voltage moves back and forth between 2/3 and 1/3 of the power supply voltage, and both of the connected comparators alternately generate reset and set signals. The flip-flop therefore delivers a nearly square-wave signal with a duty cycle t_{on}/T of about 0.5. With the chosen components, T is 1.4s.

The diode is important: Without it, a duty cycle of 0.5 is impossible to reach, because the capacitor would charge via R1 + R2 but discharge only via R2. In general (without the diode), the turn-on time t_{on} and the turn-off time t_{off} are

$$t_{on} = 0{,}69 \cdot (R1 + R2) \cdot C$$

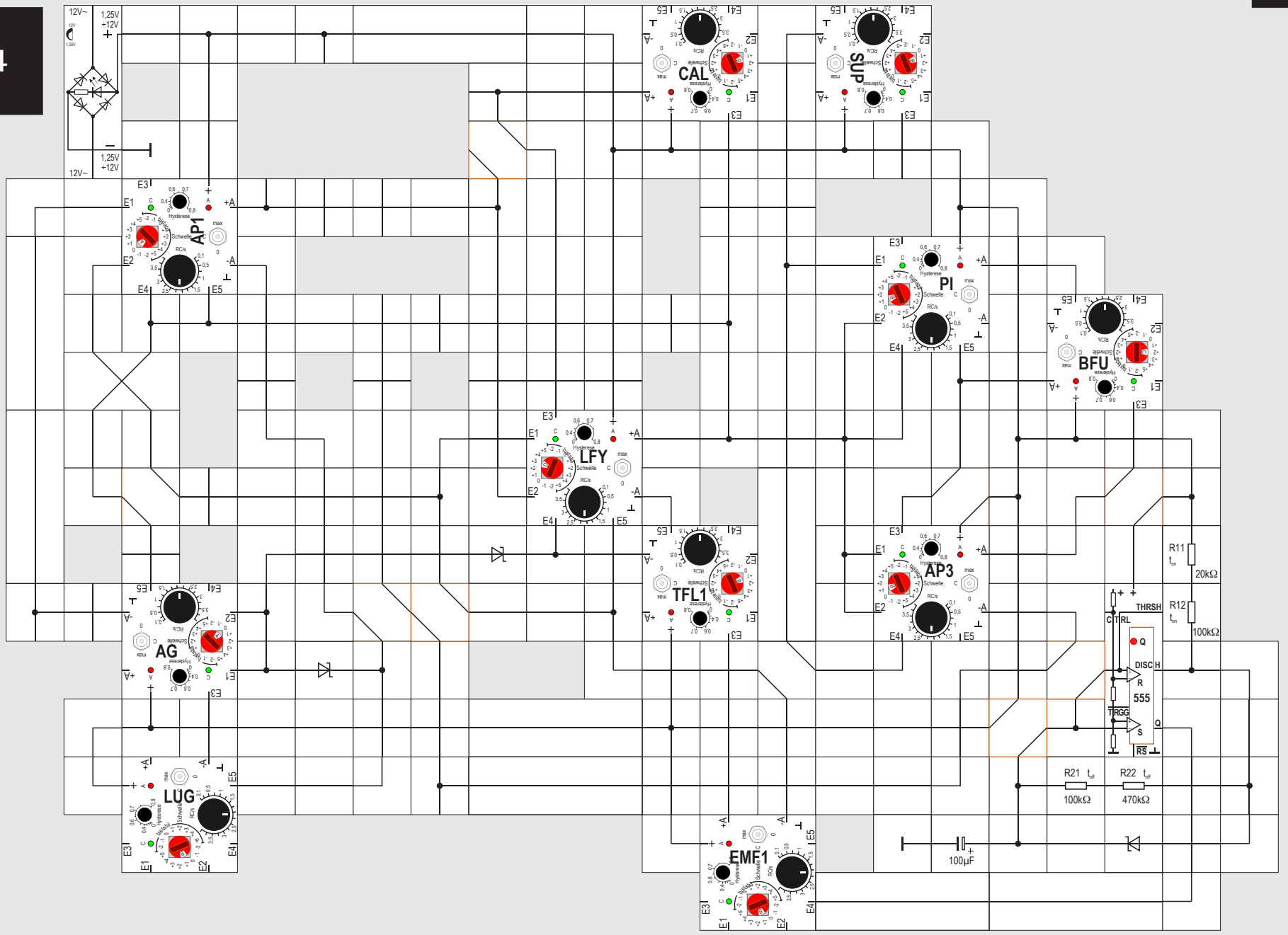
$$t_{off} = 0{,}69 \cdot R2 \cdot C$$

With the diode R2 is equal to 0 in the first formula.

To drive our network simulation, and especially to actuate **EMF1**, we increase the values of the components

R1 = (100+20)kΩ, R2 = (470+100)kΩ and C = 100µF; therefore, we need space for extra resistors to vary the time.

Then we have enough time to operate the following output of the switches. We get the setup shown on the next page (experiment 34).

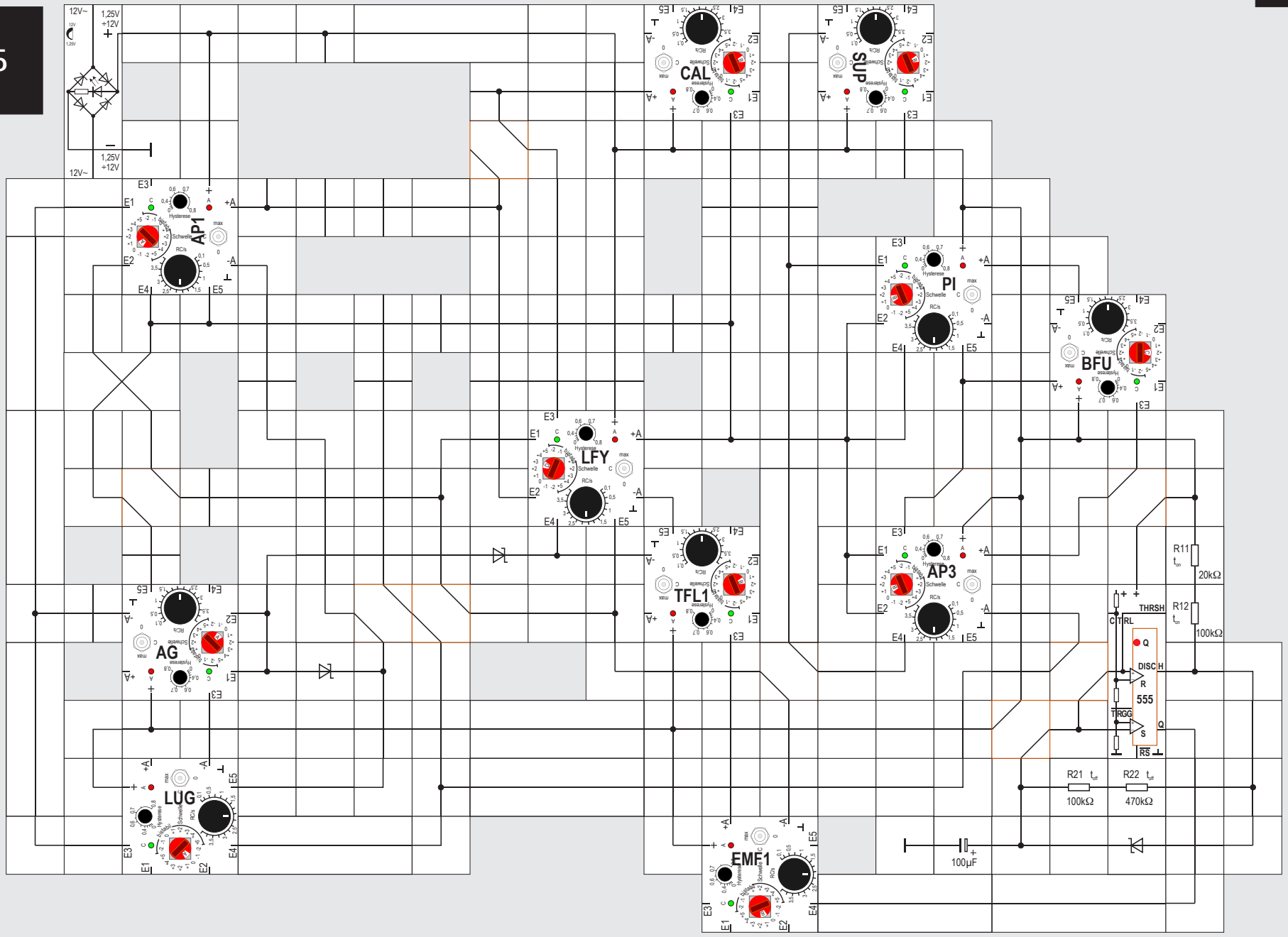

12V~
1,25V
÷12V
CAL
SUP
AP1
PI
BFU
LFY
TFL1
AP3
AG
LUG
EMF1
R11
20kΩ
R12
100kΩ
THRSH
CTRL
Q
DISCH
R
555
TRGG
S
RS
R21
100kΩ
R22
470kΩ
100µF

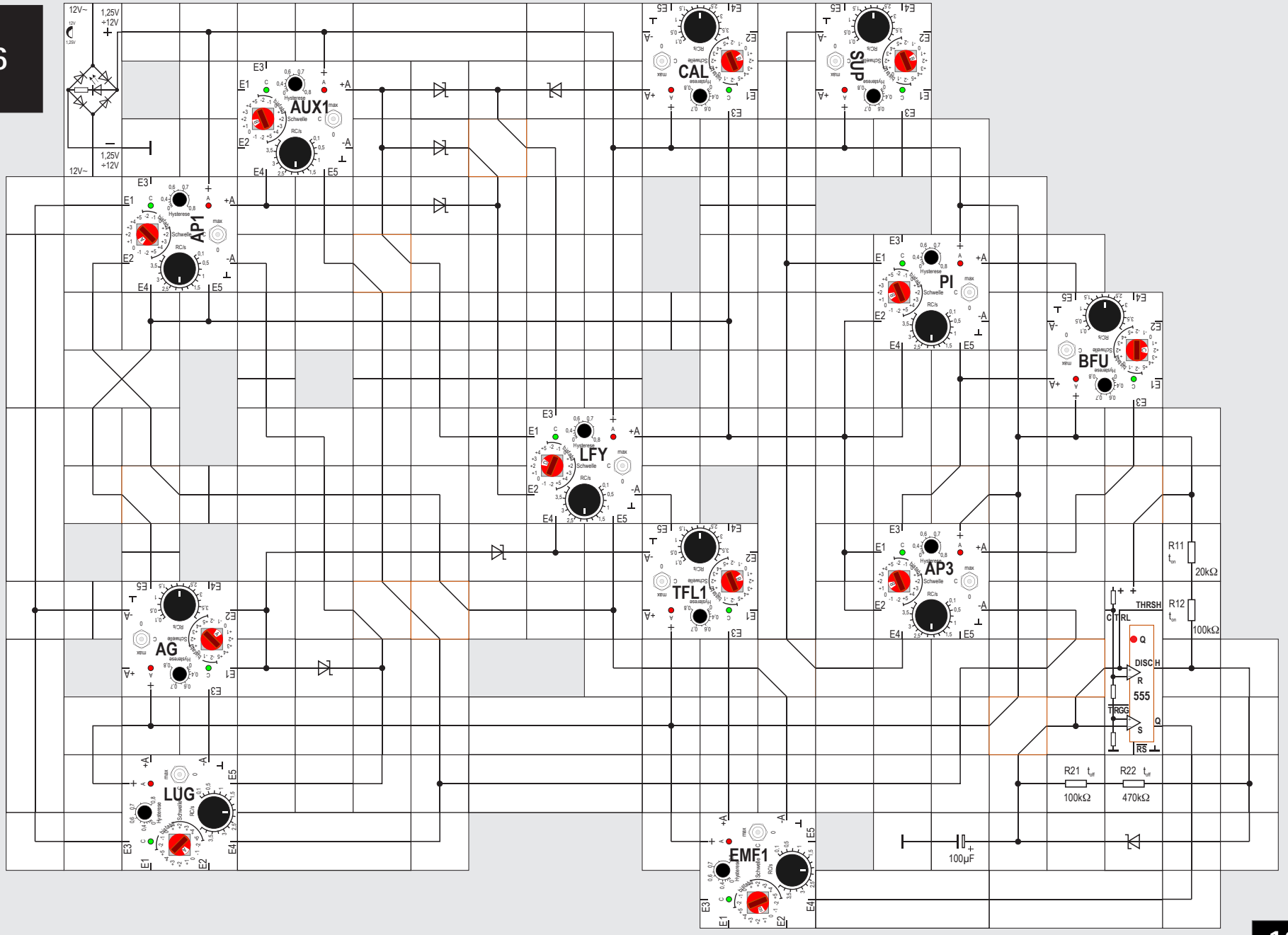
Experiment 35
Switching LUG automatically

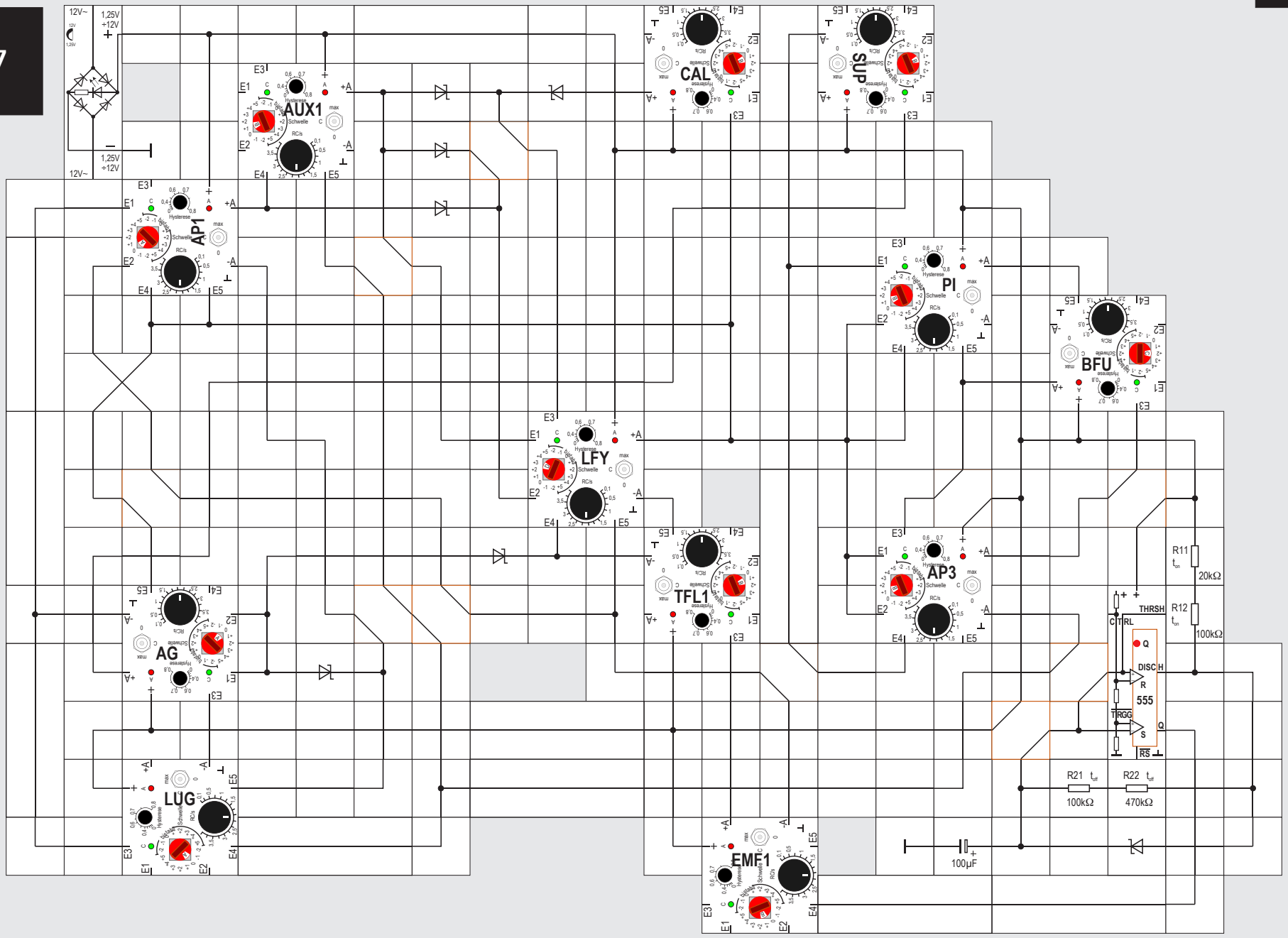
Next we consider how to save the manual switching of the **LUG** gene module by turning it on automatically. **LUG** turns itself on if its threshold is lowered from +1 to 0.
It may turn off when **AP1** is active. Therefore we connect -A of **AG** with one of **LUG**'s inputs. Since when **AP1** is no longer active later in the simulation, **LUG** would restart at an inappropriate time, we prevent this by an additional inhibition with the -A signal of **AP3**. The changed circuit is displayed on the next page. We check that everything runs as desired.

Experiment 36
Turning on LFY automatically

The following change is more extensive. We will automate the switching of the **LFY** module. Therefore we need the **OR** gene module, which we will set as an auxiliary gene and therefore call **AUX1**. Previously we noticed in an experiment that the weight of the **EMF1** signal given to **LFY** just needs to be -1 (instead of -2). By changing the wiring, we free one input of the **LFY** module.
The manual turning on overcomes the threshold +3 of the module **LFY**. The automatically generated signal of the **AUX1** module has to be connected threefold to the **LFY** module. We connect it directly to the input that just opened and get the other two by combining it with the +A signals of **AP1** and **CAL** in an OR conjunction. The needed space is already provided in the setup. The **AUX1** module is started with the +A signal of **AP1**. After adjusting the delay time, it becomes active and then starts **LFY**.

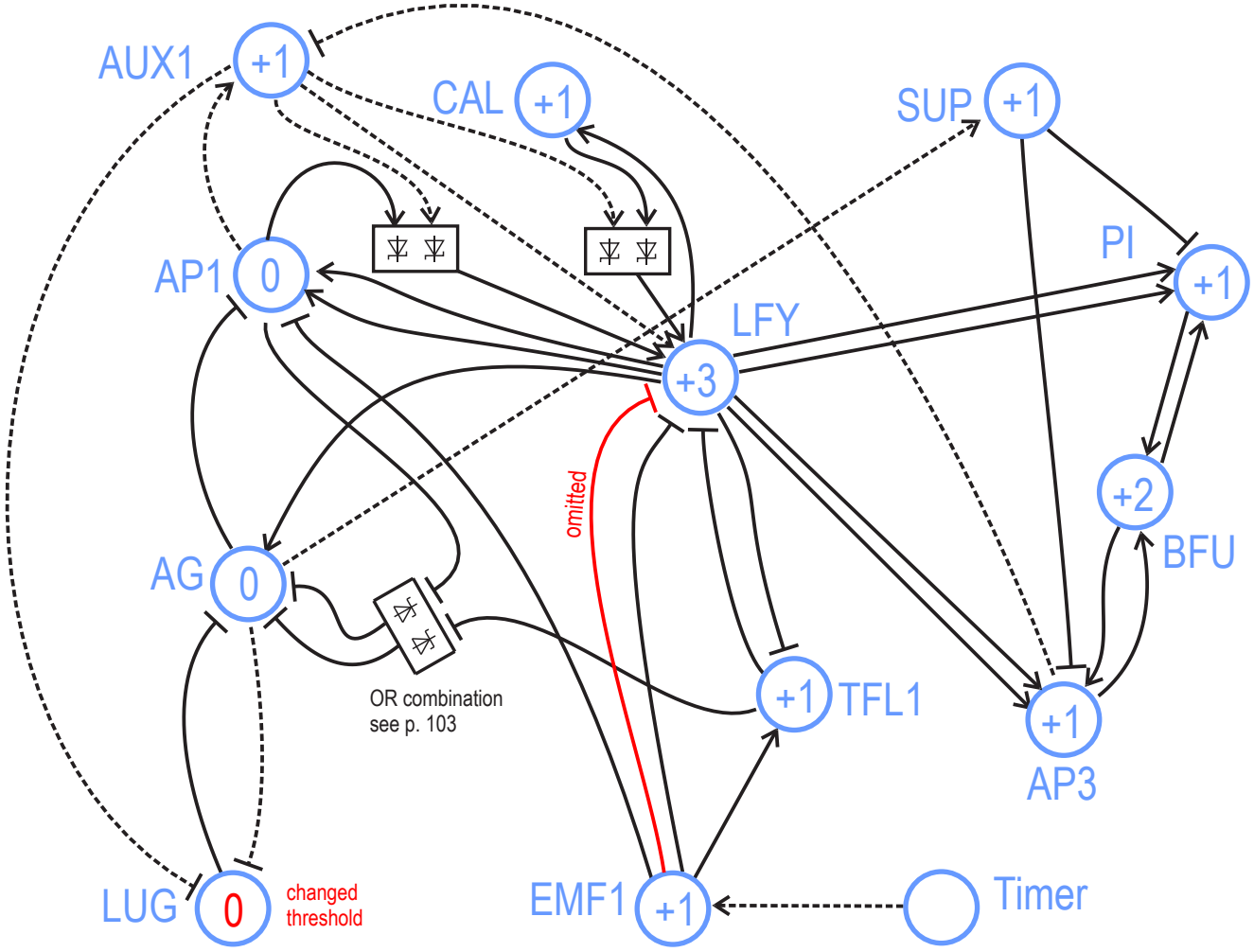

12V~
1,25V
÷12V
12V~
1,25V
÷12V
AP1
CAL
SUP
PI
BFU
LFY
TFL1
AP3
AG
LUG
EMF1
THRSH
CTRL
Q
DISCH
R
555
TRGG
S
Q
RS
R11
20kΩ
R12
100kΩ
R21
100kΩ
R22
470kΩ
100µF


12V~
1,25V
÷12V
AUX1
CAL
SUP
AP1
PI
BFU
LFY
TFL1
AP3
AG
LUG
EMF1
100µF
R11
20kΩ
R12
100kΩ
THRSH
CTRL
DISCH
555
TRGG
R21
100kΩ
R22
470kΩ


12V~
1,25V
÷12V
AUX1
CAL
SUP
AP1
PI
BFU
LFY
TFL1
AP3
AG
LUG
EMF1
THRSH
CTRL
DISCH
TRGG
555
R11
20kΩ
R12
100kΩ
R21
100kΩ
R22
470kΩ
100µF

Experiment 37
Switching SUP automatically

Finally we have to replace the manual switching of **SUP**. An appropriate signal is the +A signal of the **AG** gene module. When it is active, it activates the **SUP** gene module, too, which turns off **PI** and **AP3** and then **BFU**. With the next clock signal of the 555 timer module **ENF1** becomes active, turns on **AG** (via **TFL1**) and therefore **SUP** is turned off and the cycle restarts.
It should be clear that the timer's frequency has to match the rest of the cycle, which depends on the RC times of each gene module. If something doesn't work properly, we can correct it by adjusting the RC times. The time determining electrolytic capacitor has an especially high tolerance of 20%, so the resistor values written in the circuit diagram can vary a little. Before we show a time diagram of all signals, here is an overview with the additional (dashed) connections as they are used in the circuit.



Lectron


RC
Hyst
Thres
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150
t/s
Timer
EMF1 0.1 0.4 +1
TFL1 1.25 0.8 +1
LUG 1.25 0.8 0
AP1 3.25 0.8 0
AUX1 1.5 0.8 +1
LFY 1.0 0.8 +3
CAL 2.0 0.8 +1
PI 2.8 0.8 +1
AP3 3.0 0.8 +1
BFU 3.0 0.8 +2
AG 0.6 0.8 0
SUP 0.25 0.8 +1
11 21
60 70
109 159
11 21
60 70
109 159
14 23
63 72
111 121
16 30
114 128
29 68
127
34 48
131 145
35 49
133 147
40 54
138 152
45 84
143
43 83
141
52 90
15
74 113
16
75 113
A
B
C
carpels
sepals
petals
stamens
carpels
sepals
petals

Time diagram of the simulation of Arabidopsis flowering

The diagram shows how the single gene modules turn on and off, one after another, controlled by the timer. It is clear that first the timer signal activates EMF1, which is followed by turning the other gene modules on and off. The period of the timer is 49s. It was calculated as

$$t_{on} = 0.69 \cdot 100\mu F \cdot 120k\Omega = 8.2s$$
$$t_{off} = 0.69 \cdot 100\mu F \cdot 570k\Omega = 39.3s$$
$$T = 47.5s,$$

so it matches very well.

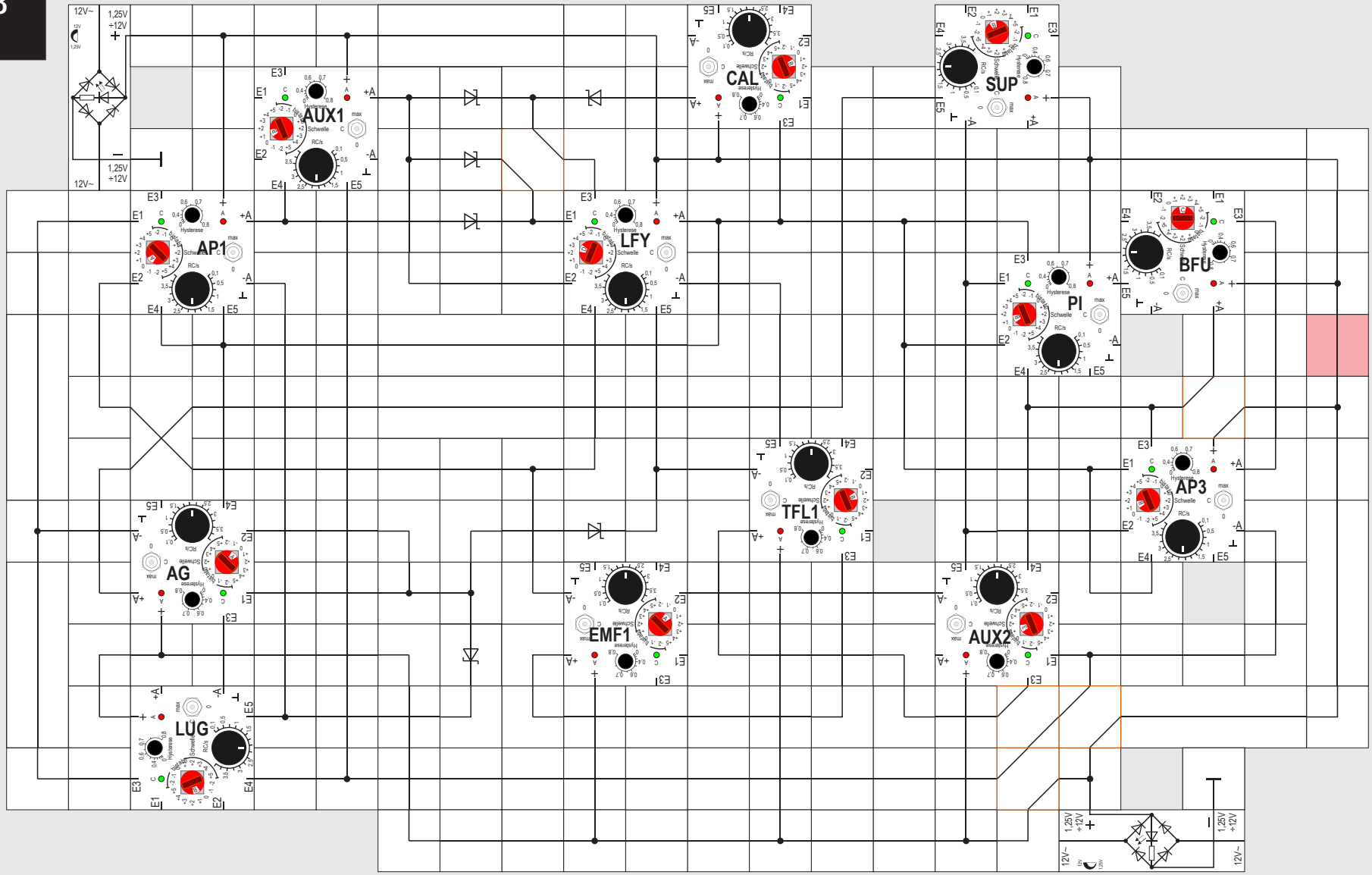
The approximate RC value of each module is listed in the second column of the diagram, the hysteresis is in the third. They are chosen more or less randomly and not optimized so they are only examples of values. We can change them cautiously and see how the time course responds. You certainly will be able to optimize them.

AG should be smaller than AP1, because by watching the green LED, we can observe that the capacitor of AP1 starts charging, but is not quite fast enough to turn it on in a timely fashion, because AG becomes active in time to inhibit AP1. This critical point is marked with a green triangle in the diagram.

Furthermore, you can find from the diagram that a full run takes two timer periods. Of course, we knew that before, because we had to turn EMF1 on manually twice.

There is a relationship between the timer period and more or less the sum of the number of times the gene modules are active. The first should not be too short; otherwise the simulation gets out of step, because the timer doesn't receive feedback on how far the rest of the sequence of switching operations has run through the course. Instead, it persistently sends its clock signals and only takes on the switching that would have to be done manually.

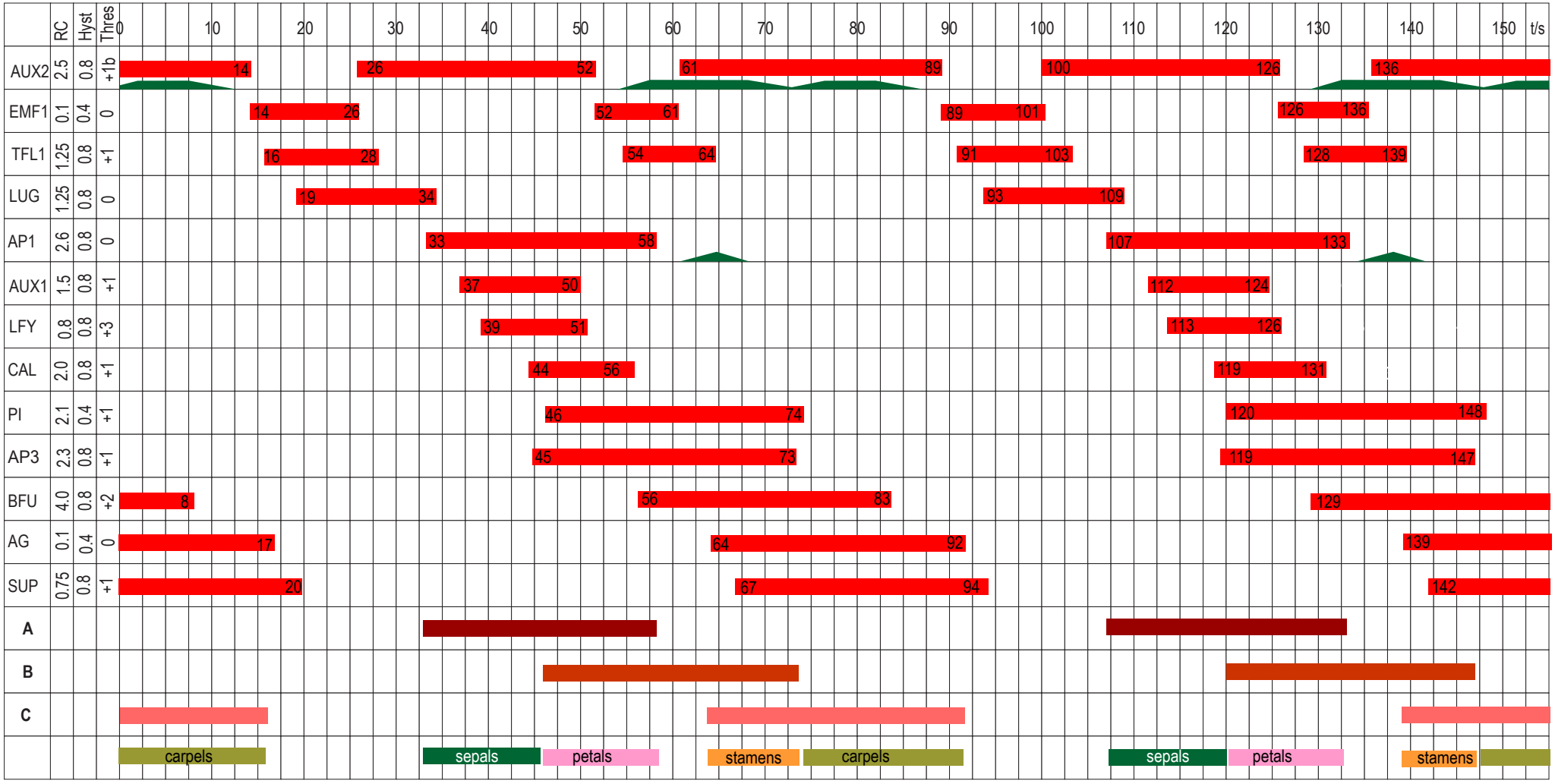
Even though this is very nice, we might ask if it is possible to let the cycle start itself without the timer. If we manage to create a signal starting EMF1 in the otherwise unchanged and satisfactorily running circuit, and it has a similar timing to the current one, the rest of the course should run as before undisturbed.


12V~
1,25V
÷12V
+
–
1,25V
÷12V
12V~
AUX1
CAL
SUP
AP1
LFY
BFU
PI
AP3
TFL1
AG
EMF1
AUX2
LUG
Schwelle
RC/s
Hysterese
max
+A
-A
E1
E2
E3
E4
E5

Experiment 38
Simulation of Arabidopsis flowering without a timer

We interrupt the connection between the timer and the **EMF1** module and consider how to activate the last one. The simplest way is to set its threshold to 0, so it is activated without a trigger. That works without trouble.
Afterward, it has to be turned off after 10 seconds or so. We need a signal from the remaining circuit that is able to perform that. We could use the -A signals from **TFL** and **LUG**. Unfortunately they do not last long enough, so **EMF1** quickly becomes active again. The same is true for **AG** and **SUP**.
So we are reduced to adding another gene module (the 13th): **AUX2**. This module will be activated by **TFL1** and turn off **EMF1** with its -A signal. So it will not be not turned off itself in the sequence **EMF1** **TFL1** **AUX2**, releasing **EMF1** again, we set its threshold to +1 bipolar; without an incoming signal, it remains turned on and keeps on inhibiting **EMF1**.
We have to look for a -A signal that turns off **AUX2** and consequently turns on **EMF1** for a second time, because a full simulation cycle needs an active **EMF1** signal twice. The time diagram shows that **AP3** can do it. It turns off **AUX2** and turns on **EMF1**. Thus the second half of the cycle can begin.
To carry out a process like this, **EMF1** would have to be turned off again by an active signal of **AUX2** shortly afterward. The reactivation of **AUX2** is done by the +A signal of **BFU**. When **BFU** and **AP3** are deactivated, the -A signal of the **SUP** gene module is finally able to turn off **AUX2**, which leads to the activation of **EMF1** again so the cycle can start again.
We have to do some reconstructing in the right part of the setup. Otherwise, we can't lay in connections to the **AUX2** gene module. The amended connection diagram supports the alterations. We now must provide 13 gene modules with electrical power, but the power supply module is at its limit. If it overheats, it might turn off (reversibly). A second power supply module in the setup would do no harm, and would share the burden. Before it is put into the setup, don't forget to omit the pink connection module!

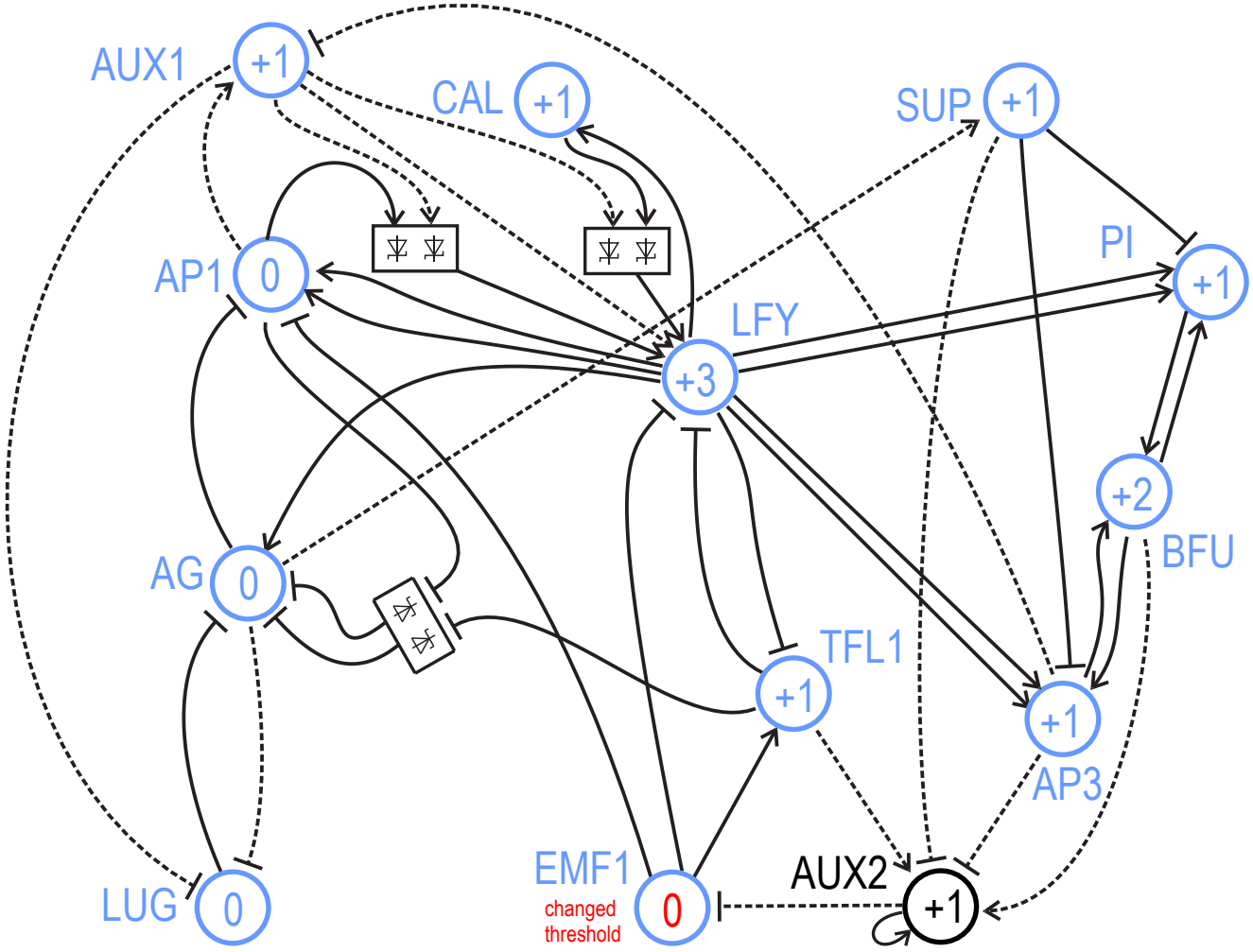
Lectron

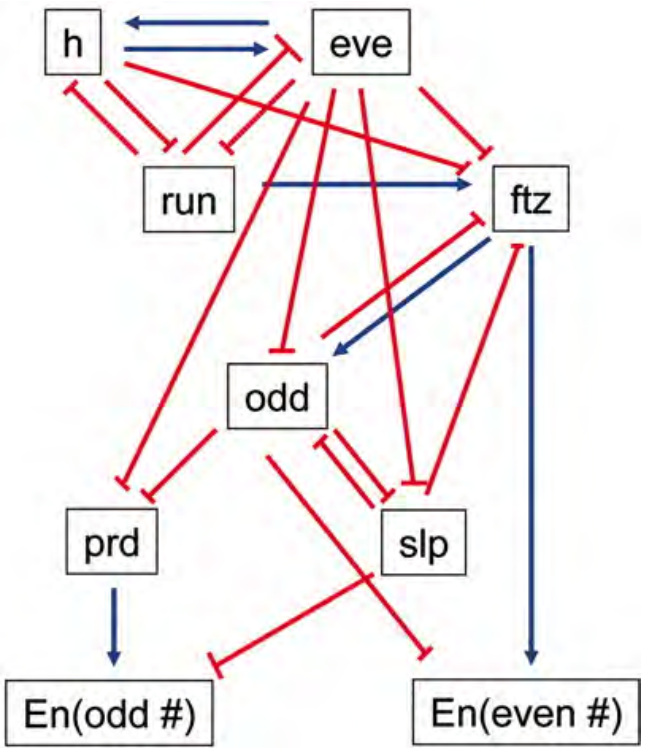

RC
Hyst
Thres
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150
t/s
AUX2 2.5 0.8 +1b 14 26 52 61 89 100 126 136
EMF1 0.1 0.4 0 14 26 52 61 89 101 126 136
TFL1 1.25 0.8 +1 16 28 54 64 91 103 128 139
LUG 1.25 0.8 0 19 34 93 109
AP1 2.6 0.8 0 33 58 107 133
AUX1 1.5 0.8 +1 37 50 112 124
LFY 0.8 0.8 +3 39 51 113 126
CAL 2.0 0.8 +1 44 56 119 131
PI 2.1 0.4 +1 46 74 120 148
AP3 2.3 0.8 +1 45 73 119 147
BFU 4.0 0.8 +2 8 56 83 129
AG 0.1 0.4 0 17 64 92 139
SUP 0.75 0.8 +1 20 67 94 142
A
B
C
carpels
sepals
petals
stamens
carpels
sepals
petals
stamens

The above time diagram (note the changed settings!) shows in detail how the switching operations work in a cyclic way. We recognize that the actual period in comparison to the circuit with the timer has become shorter. That means we could have alternatively adjusted a shorter period for the experiment with the timer. Additionally, both **EMF1** signals, which occur in one simulation cycle, do not have exactly the same length.

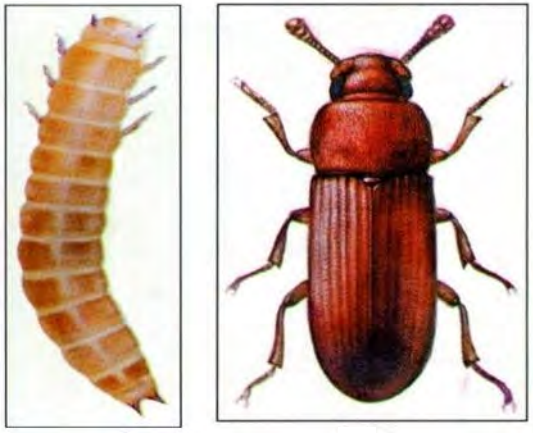
Besides the critical time adjustment of **AP1** and **AG**, which still remains, a second one occurs at the **AUX2** module. We recognize by its green LED that from second 68, the capacitor starts discharging, because two signals, **SUP** and **AP3**, are inhibiting **AUX2**. **AP3** just turns off in time so **AUX2** remains active in between. By changing the times carefully, we can optimize the whole thing. The circuit has no stable state; in principle it is a complex oscillator that runs cyclically through a few of its 2^{13} possible states.

Compared to the yeast circuit we need many additional (dashed) interference connections. A cycle requires two »triggers« with very similar start conditions; however, the circuit must memorize these little differences.



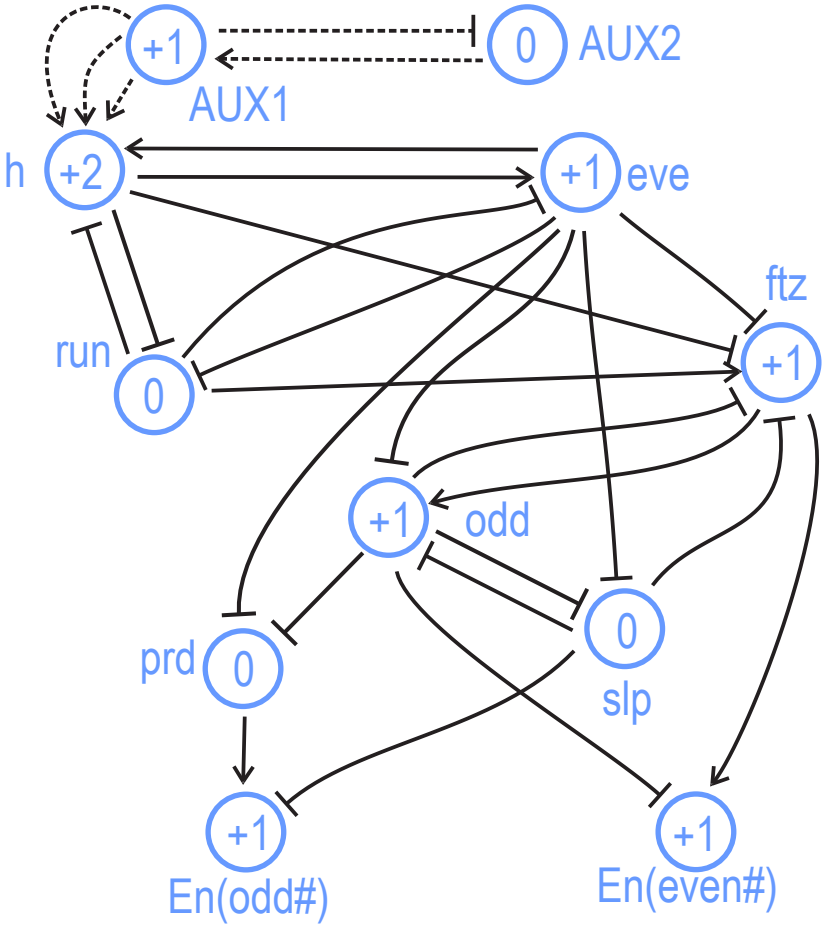


from: Choe et al. [16]



Grub Beetle

Tribolium castaneum



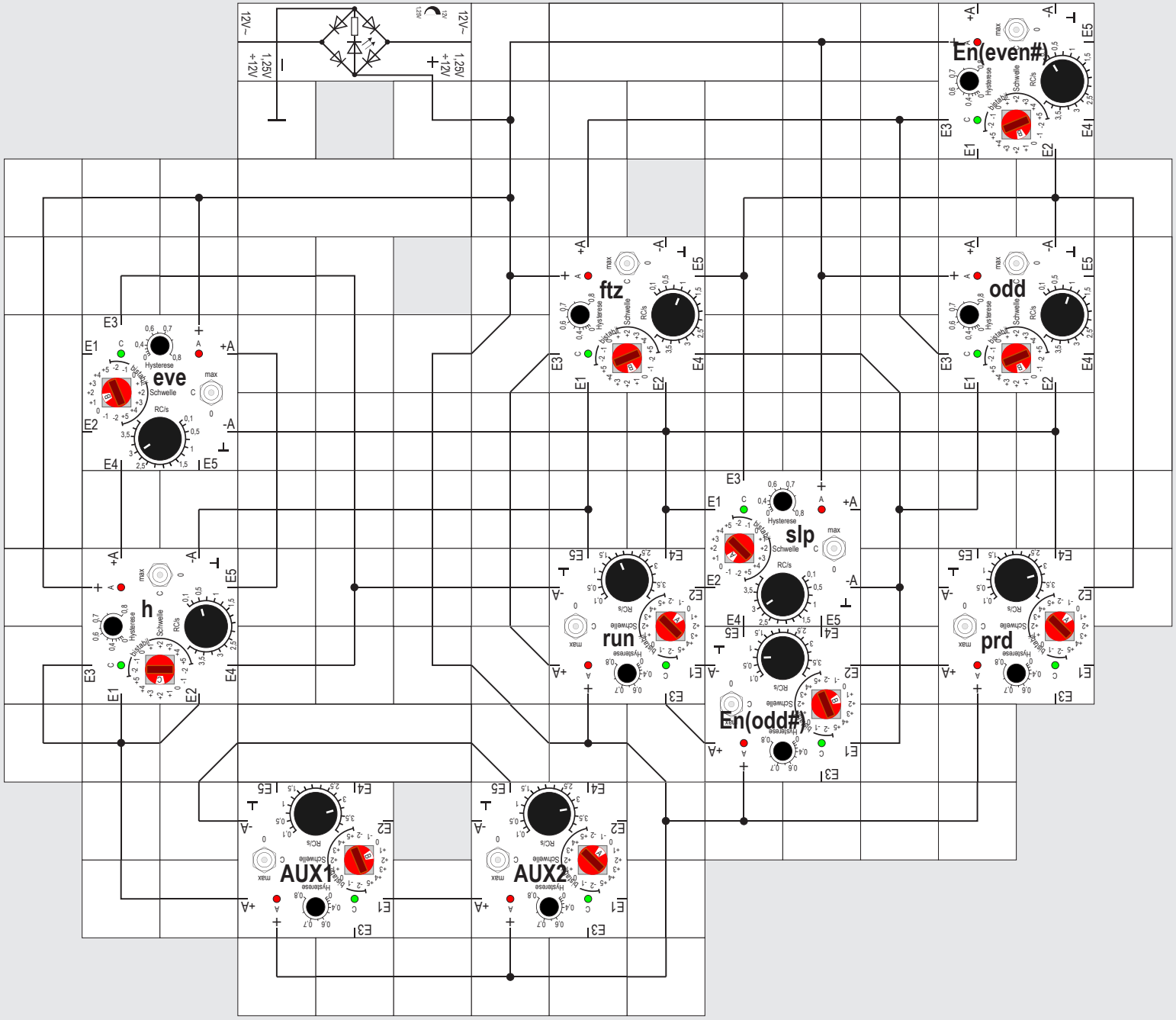
Experiment 39

Simulation of segment development of the red flour beetle larvae

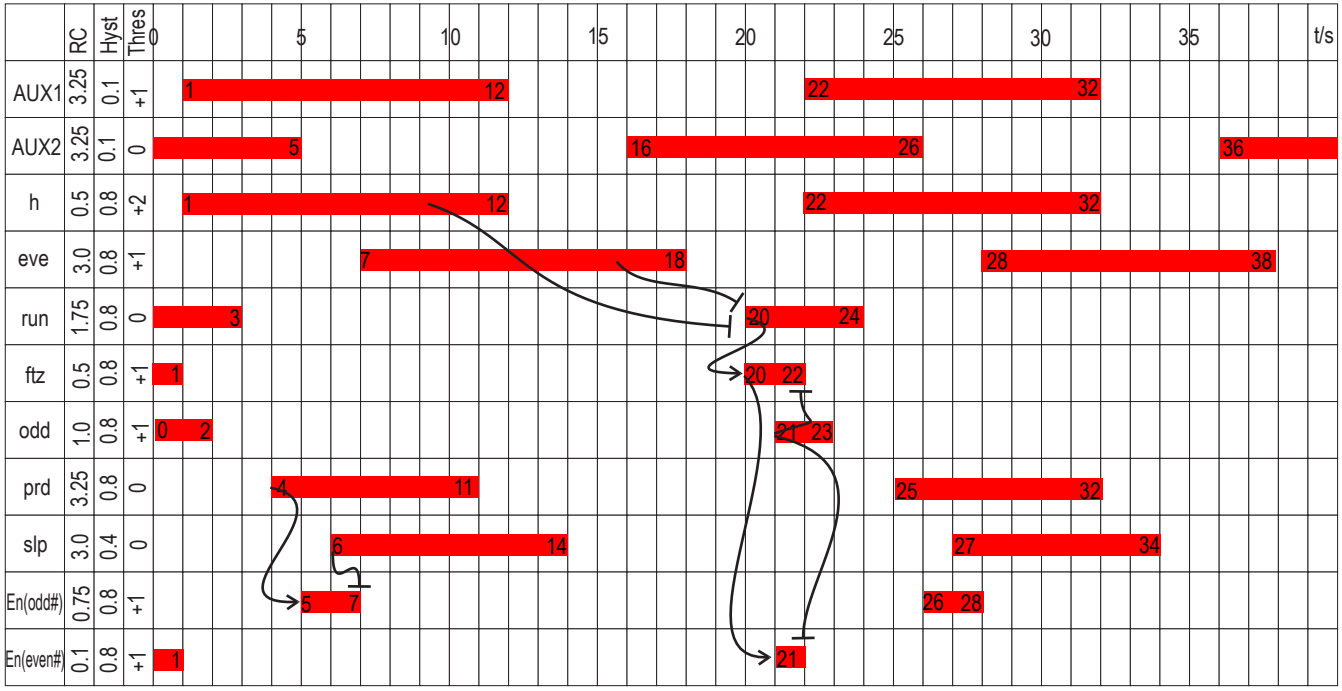
Now we turn to an example in developmental biology, because the interaction of cells during the development of a multicellular organism is often controlled by a rhythmic clock inside the cells. The following example is such a clock that takes part in the segmentation of insect embryos [16]. It requires nine gene modules and describes how pairs of segments are formed in the development of the larvae of the red brown flour beetle (*Tribolium castaneum*).
According to Wikipedia,

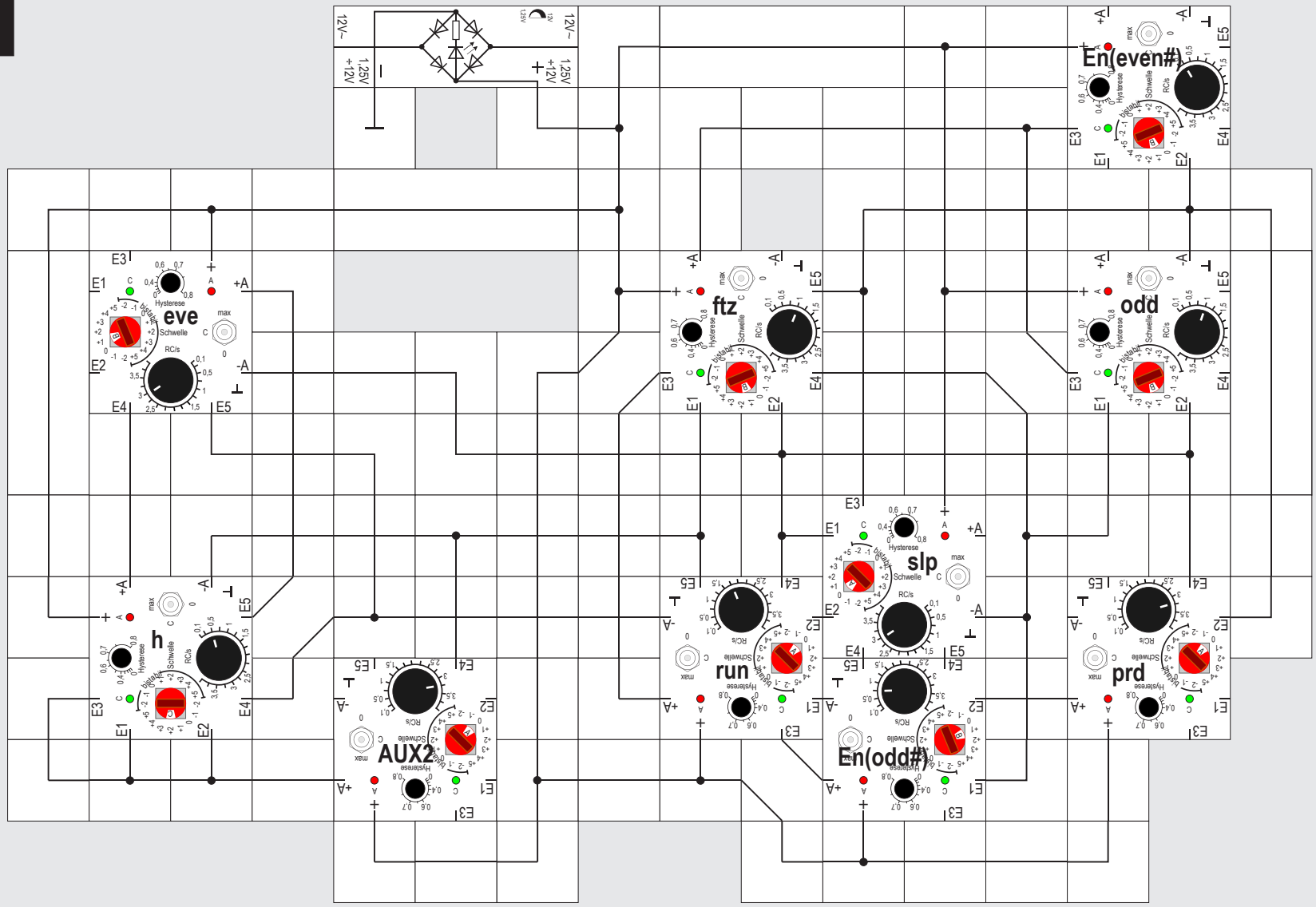
»the red flour beetle attacks stored grain and other food products, including flour, cereals, pasta, biscuits, beans, and nuts, causing loss and damage. It may cause an allergic response but is not known to spread disease or cause damage to structures or furniture. The United Nations, in a recent postharvest compendium, estimated that *Tribolium castaneum* and *Tribolium confusum*, the confused flour beetle, are the two most common secondary pests of all plant commodities in store throughout the world.«

This beetle is about to become an important genetic model organism. On March 24, 2008, it became the first beetle with a sequenced genome; it contains 16,000 genes, several hundred of which have not been found in another model animal, the fruit fly. Scientists hope to find insights in embryonic development, metamorphosis, evolution of body diversity, and control of harmful organisms using the flour beetle as a model system.
Within the analysis of the examined gene network of this prominent beetle, neither weights of input signals nor thresholds of the gene modules have been noted yet; we first notice that the genes **h** and **eve** are coupled ++. That means they are either both turned on or off after a short time. If they are active, they inhibit all genes in the middle of the network. None of these genes, **run**, **slp** or **prd**, receive activating signals from other genes. To become active, their thresholds must not exceed 0. The two lower genes **En(odd#)** and **En(even#)** don't emit signals. In the end, they are only responsible for a larva to develop a segment with an even or odd ordinal number. We exJust like in the previous simulations, we want it to run periodically, even though this doesn't happen in reality. The larva, of course, can't form legs forever. Therefore it is easiest to connect an external clock with the network to periodically turn h and eve on and off since they will not do spontaneously. Because we have gene modules left this time, we use an oscillator circuit with two modules **AUX1** and **AUX2**, whose on and off times can be adjusted separately. On the following page, a possible compact setup is displayed. Because unused inputs of different gene modules can be connected to each other, **run**, **slp** and **En(odd#)** are close neighbors. The unwanted connection between **En(odd#)**'s -A output to **run**'s E1 is compensated for by the connection between **En(odd#)**'s +A output to its E3. To simplify matters, the thresholds of **eve**, **ftz**, **odd**, **En(odd#)**, **En(even#)**, and **AUX1** will be +1 at first. To make the oscillator work, **AUX2**'s threshold must be 0. The external clock can only switch effectively if it affects **h** with weight 3, which will then have a threshold +2. The detailed circuit is shown above.


12V~
1,25V
+12V
En(even#)
ftz
odd
eve
slp
h
run
prd
En(odd#)
AUX1
AUX2

When the setup is complete and checked, even with the thresholds determined by the initial considerations, you have a vast number of possibilities for adjusting the time constants and hysteresis of each module. To avoid having the whole thing proceeding completely without a plan, there is a time diagram on the right showing successful activation of **En(even#)** and **En(odd#)** one after the other, even though it is only for a short time. This at least proves that our thresholds can't be completely wrong. The adjustments of RC and hysteresis on the left of the diagram are only reference values, though. We can't expect to get exactly this time diagram with those adjustments. You will see that this network requires a lot of patience and a great deal of sensitivity to get the desired course. Because of the strong networking, even tiny changes to a time constant, you shouldn't change more of them at a time anyway, since this often changes things unexpectedly in very different corners of the network. For additional help with the adjustments, the most important dependencies of the signals are noted in the diagram. For instance, **En(odd#)** is activated by **prd** and deactivated by **slp**. Similar considerations apply to **En(even#)**: **ftz** activates; **odd** inactivates. Since **AUX1** and **h** always switch at the same time, we can omit the **AUX1** module and build the oscillator with **h** and **AUX2**. We then get the simpler setup shown on the next page.

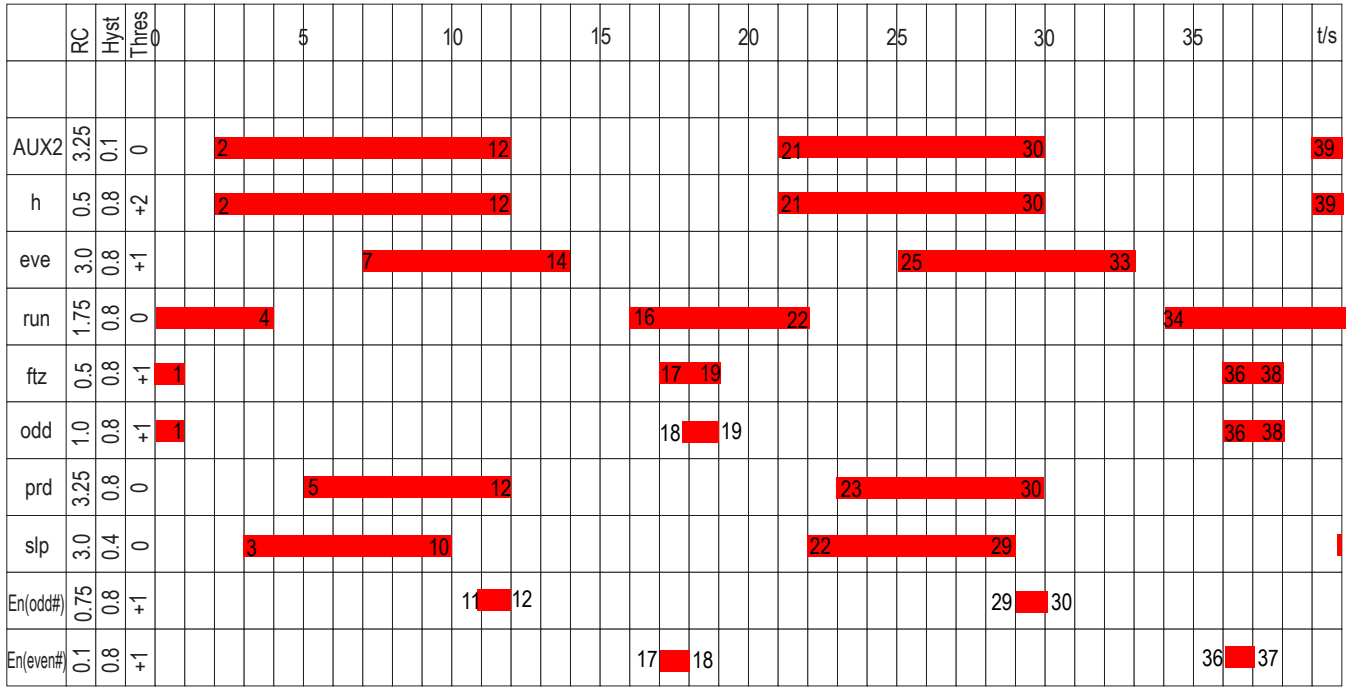



12V~
+12V
En(even#)
eve
ftz
odd
h
slp
run
prd
AUX2
En(odd#)

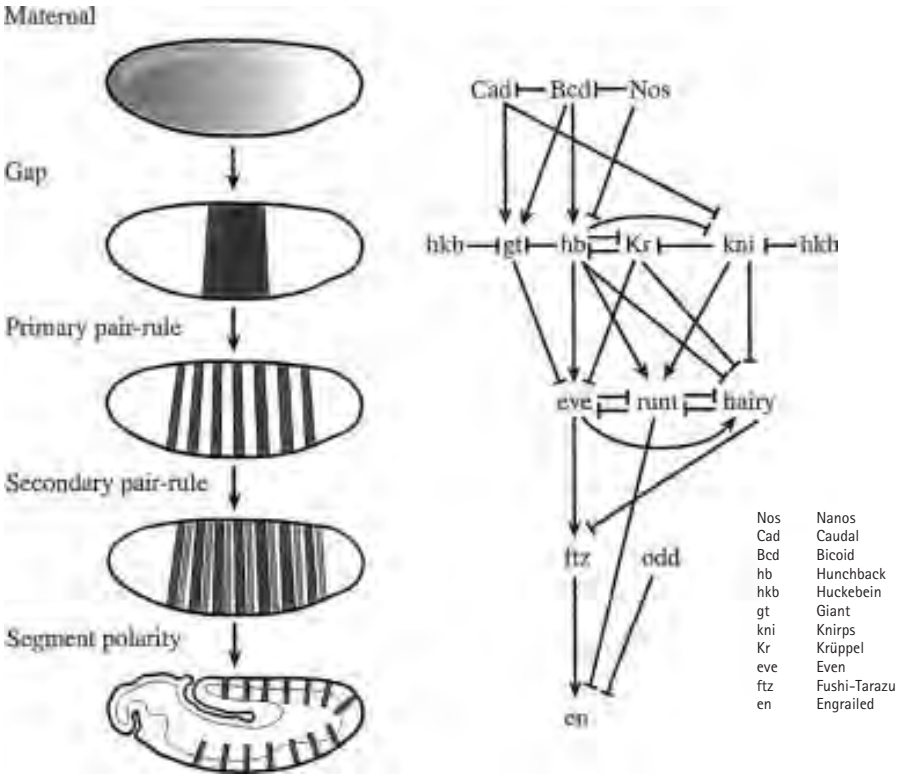
Experiment 39A

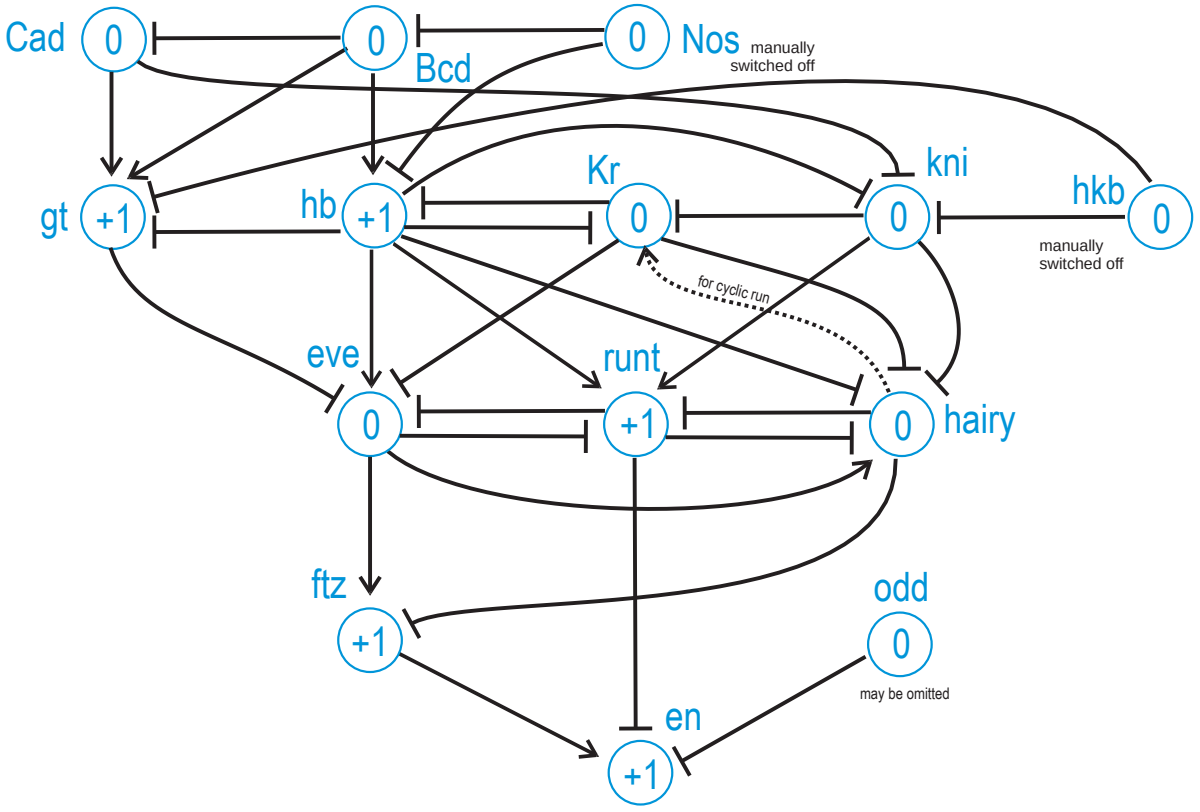
Segment development of the red beetle grubs Alternative setup

Like before, we have to be patient and careful with the time adjustment of RC and the hysteresis to get the time diagram shown to the right. The adjustment values in the columns are just estimates; we take them as prior settings and try step by step to get better results by gently altering single adjustments of the modules. You will be surprised at the strange signal sequences you get when trying to optimize the process. This more or less randomly chosen example of a gene network simulating a process in developmental biology may already be too complicated to get a good immediate understanding of its control.



Lectron





From: Carroll et al. [17]

Experiment 40
Simulation of segment development of the common fruit fly, ***Drosophila melanogaster***

After the example of the exotic red beetle grub, we want to go through the regulation network of the grub of the small but well-known common fruit fly *Drosophila melanogaster* [17]. The complete ~3mm long insect is particularly known for being attracted to overripe fruit. Nevertheless, for almost a century, it has been a model organism for developmental biologists and geneticists. Why, of all insects, the common fruit fly?

This is mainly because of its short life cycle of less than two weeks and the simple and cheap maintenance of the organism; even if you need a lot of flies for experiments, that's no problem. In addition, there are thousands of *Drosophila* mutants, and the sequence of its genome has been known since the year 2000.

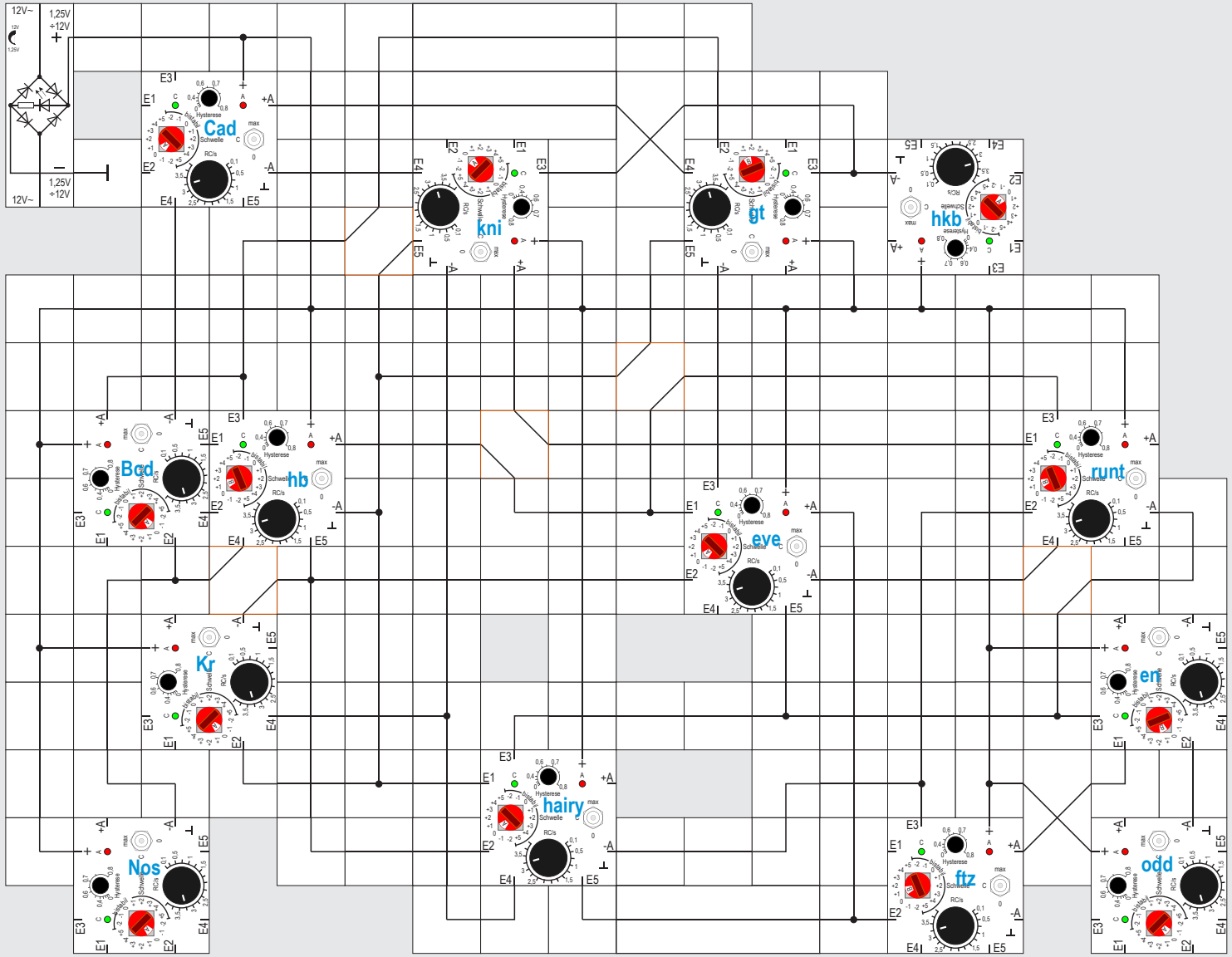
By now, *D. melanogaster* is so well-known that a biology class that doesn't mention it is hardly imaginable. This fly is used as an example of the rules of heredity in genetics and to explain the basic genetic control mechanisms in embryonic development.

Drosophila can very precisely be used to examine how a complex organism arises out of a simply fertilized egg through an embryo and an organic process of development. That's what the regulation network is all about. The left panel of the figure on the left side of the previous page allows us to recognize how structures in the early embryo of the *Drosophila* are controlled by genes in time and space. The right figure shows the detected gene regulation network that produces this structure.

All in all, 14 genes (we already know some) that are activated on five levels of hierarchy are responsible. Since we only have 13 gene modules, we must decide which one we can omit if we do not want to buy another one.

At a glance, we recognize three genes, **Nos**, **hkb**, and **odd** that receive no input signals but emit only inhibiting signals. These gene modules may get a threshold 0 at the most; otherwise nothing will happen when we turn on the power supply. Thus they get active and interfere with the course by their inhibiting signals. Then nothing will happen anymore. The most obvious one to omit is **odd**. When **odd** is active, only **en** is inhibited. In the first instance, **hkb** and **Nos** should be turned off manually by moving their toggle switches to 0, or even better, don't get any power. As these very modules are situated at the corners of the setup, we can easily contrive this. The power supply has to power a lot of modules and is therefore working at its limit anyway; this way, it will not get as hot.

We know the activating and inhibiting signals of this network, but the thresholds of the gene modules are not stated. Generally they should be adjusted with low numbers so the input signals can exceed the thresholds and activate them.


12V~
1,25V
÷12V
+
−
12V~
1,25V
÷12V
Cad
kni
gt
hkb
Bad
hb
eve
runt
Kr
en
hairy
Nos
ftz
odd

We choose the threshold 0 for **Bcd**, the highest module without an input signal in the hierarchy; therefore, it activates itself without any help from other modules. The same applies to **Cad**. Moreover, it needs a smaller RC value than **Bcd**, so it can be active just for a short period of time before it gets inhibited by **Bcd**.

At the next level, **gt** and **hb** can get the threshold +1 because they get an activating signal. In contrast, **Kr** and **kni** only get inhibiting signals, so a threshold of 0 is a good choice.

The threshold is not at all important for the manually controlled **Nos** module and the **hkb** module one level lower; **eve** and **hairy** mostly get inhibiting signals. Therefore, we choose the threshold 0; **runt** may be able to reach the threshold +1. After all, there are **ftz** and **en** left over. As a start, we try the threshold +1 and see if an activation will take place.

A possible space-saving circuit is shown above with 14 gene modules; as mentioned, if required, we can turn off or even omit **Nos**, **hkb** or **odd**.

When looking at the interacting relations of the net more closely, you probably see that there are some pairs of genes that inhibit each other as toggle switches (see experiments 9 and 14). Such pairs are **hb**↔ **Kr**, **eve**↔ **runt**, **runt**↔ **hairy**, and indirectly (via **Kr**) **hb**↔ **kni**. That a gene like **runt** is in more than one toggle relationship and that the toggle switches are coupled doesn't make it easy to recognize how the process will work.

If we apply the power supply to the setup as shown on the previous page (**Nos**, **hbk**, and **odd** are deactivated), it is surprising how the network behaves with the chosen thresholds and with more or less random RC and hysteresis adjustments. First, all gene modules become active; 40 seconds or so later, four genes (**Bcd**, **hb**, **eve**, and **hairy**) are permanently turned on. You will find the process diagram with all the information about thresholds and hysteresis as well as the RC adjustments on the next page.

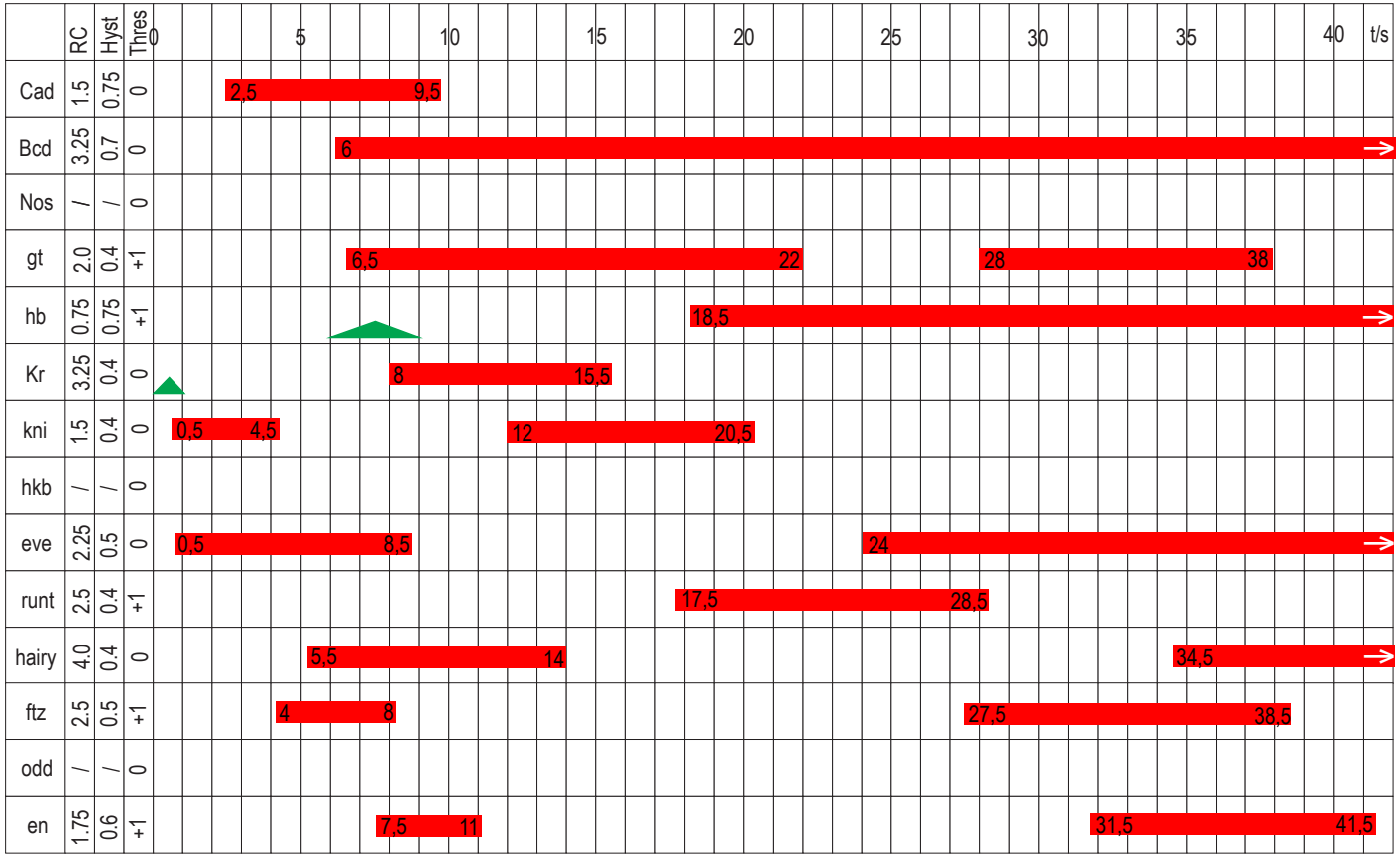
If we turn on the power supply and set the **Nos** module switch to »C« and its threshold to »0«, we get a shorter run in which **hb** isn't even active, and in the end, **Cas**, **gt**, and **Kr** remain active.

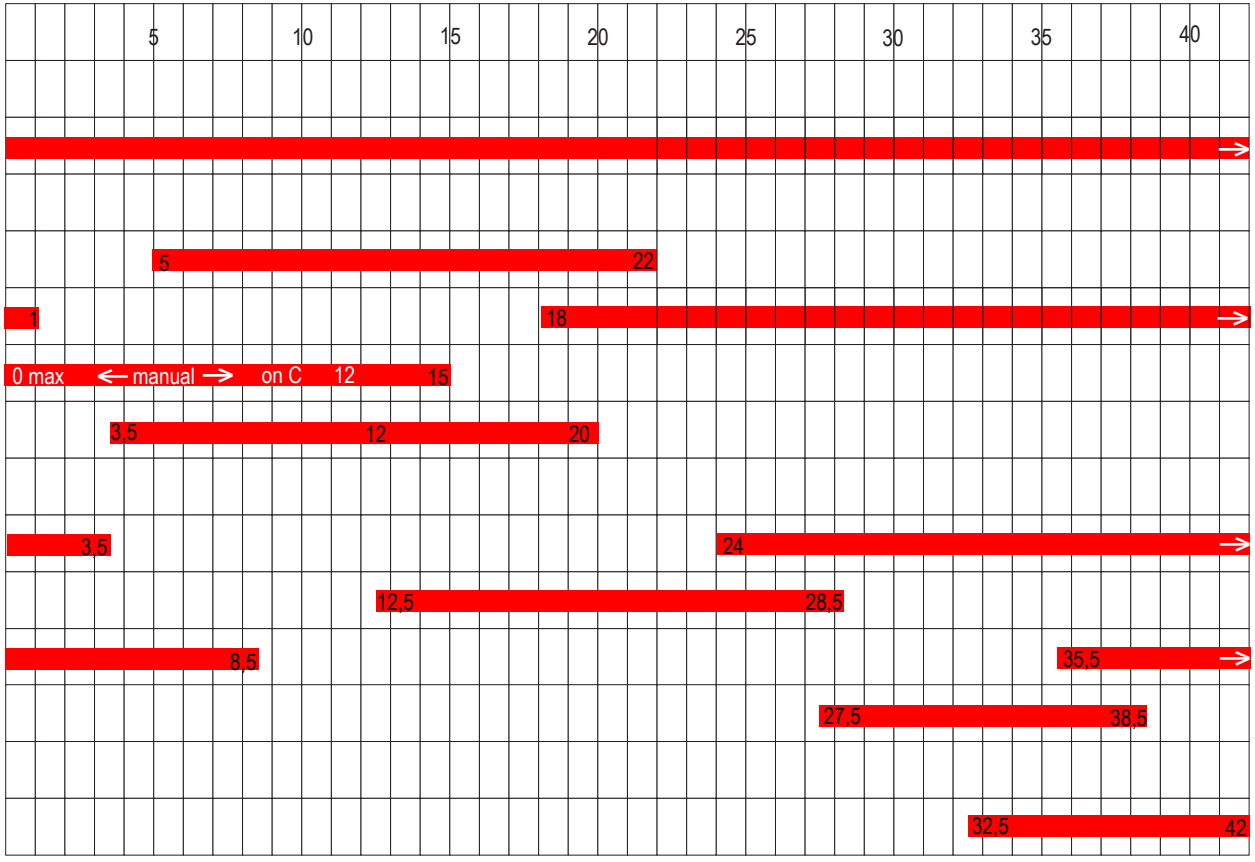
If in the beginning the switch of the **hbk** module is at »C« (with its threshold at »0«), there is again a shorter but different run in which **runt** isn't active and finally **Bcd** and **Kr** remain active.

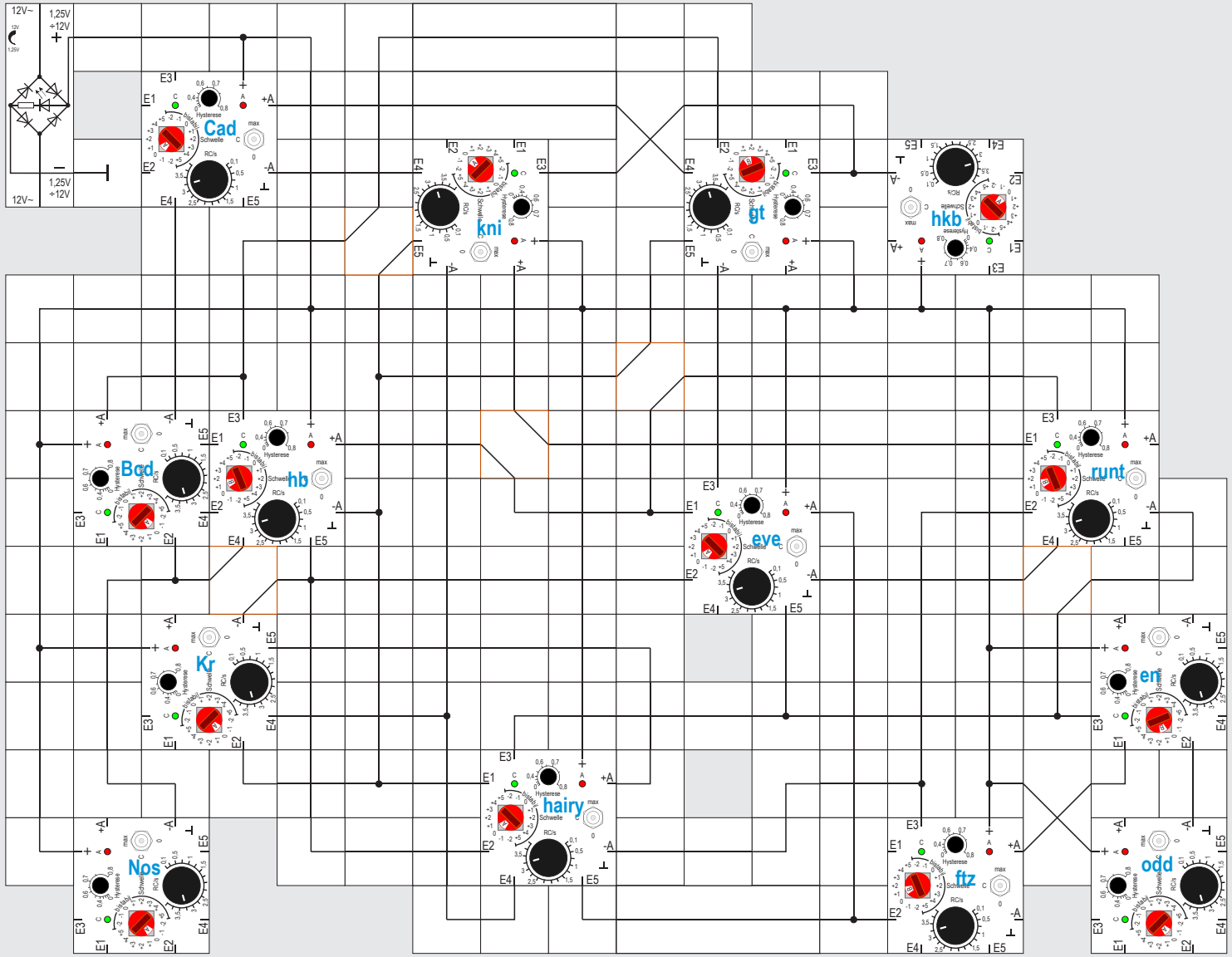
The combination of both (**Nos** and **hbk** switches at »C«) leads to a run where neither **hb** nor **runt** become active at all, but in the end **Cad** and **Kr** remain active.

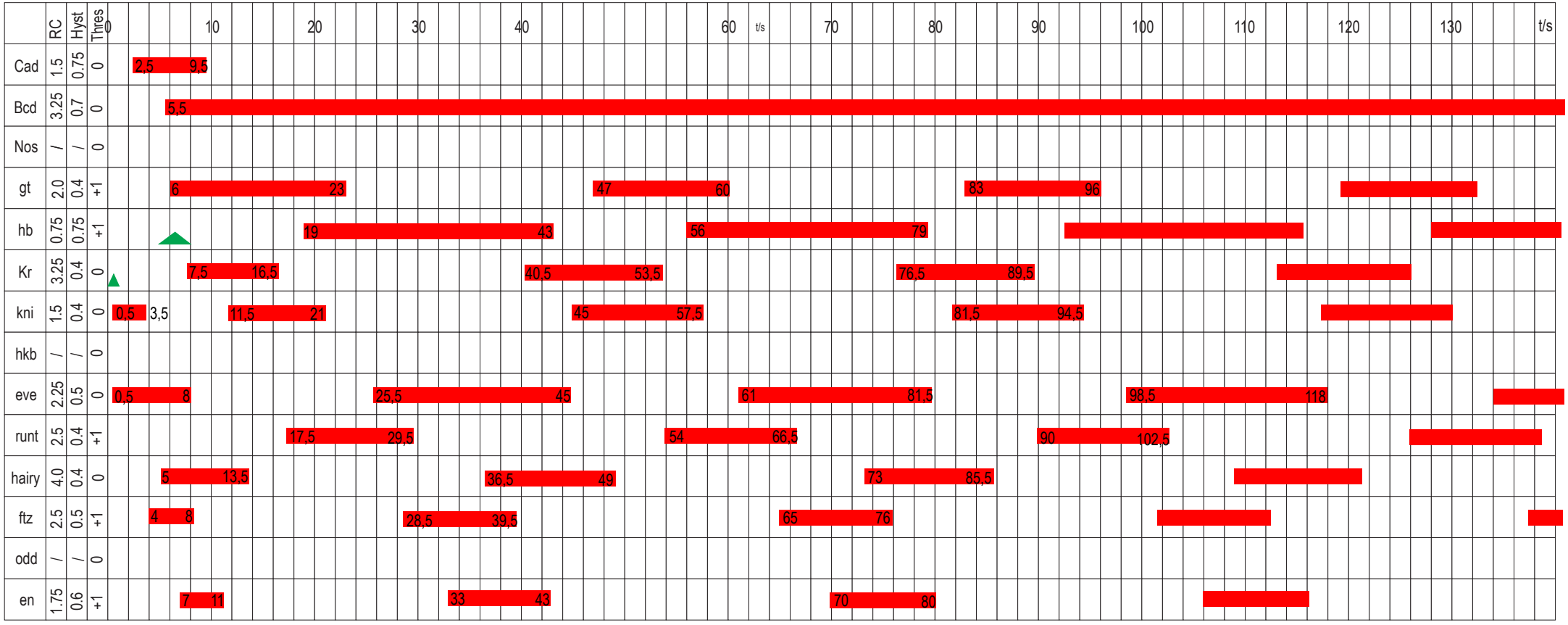
Let's return to the first run where **Nos** and **hbk** were turned off, so the stable state is reached after ~40 seconds. If we now turn the **Kr** switch manually to »max« and after ~12 seconds to »C«, we get another new run with a time diagram that is also given. It is very similar to the first, even if there are some differences; the stable state is reached after about 40 seconds, and at the end, **Bcd**, **hb**, **eve**, and **hairy** remain active again.

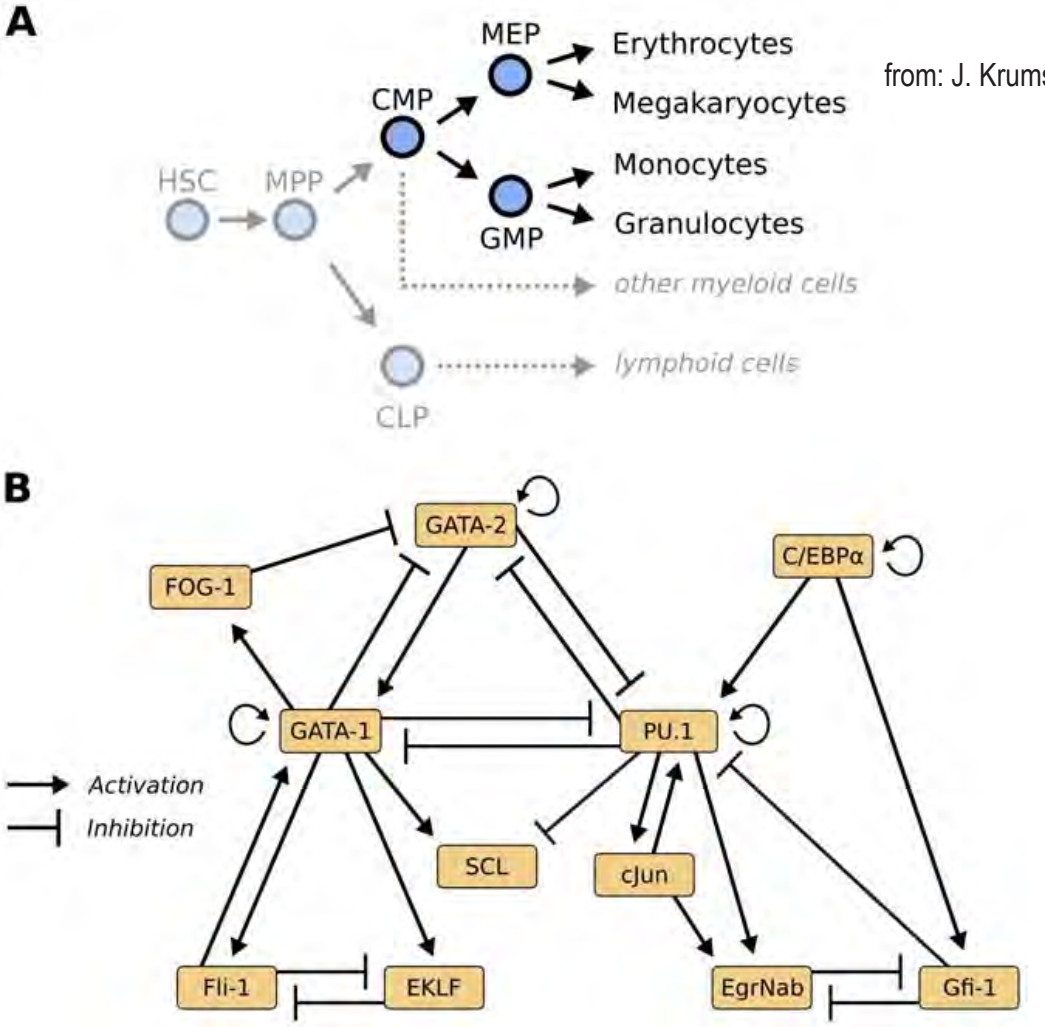
Manual operation of the **Kr** switch is not necessary to start the process; the **hairy** module is able to do that automatically with its +A signal: We connect the output with the input of the **Kr** module (see the other circuit on page 130) and now it runs cyclically. However, this connection doesn't exist in nature; it is just introduced to demonstrate its effects.


RC
Hyst
Thres
0
5
10
15
20
25
30
35
40
t/s
Cad 1.5 0.75 0
2,5
9,5
Bcd 3.25 0.7 0
6
Nos / / 0
gt 2.0 0.4 +1
6,5
22
28
38
hb 0.75 0.75 +1
18,5
Kr 3.25 0.4 0
8
15,5
kni 1.5 0.4 0
0,5
4,5
12
20,5
hkb / / 0
eve 2.25 0.5 0
0,5
8,5
24
runt 2.5 0.4 +1
17,5
28,5
hairy 4.0 0.4 0
5,5
14
34,5
ftz 2.5 0.5 +1
4
8
27,5
38,5
odd / / 0
en 1.75 0.6 +1
7,5
11
31,5
41,5


5
10
15
20
25
30
35
40
5
22
1
18
0 max
manual
on C
12
15
3,5
12
20
3,5
24
12,5
28,5
8,5
35,5
27,5
38,5
32,5
42


12V~
1,25V
÷12V
+
−
12V~
1,25V
÷12V
Cad
kni
gt
hkb
Bad
hb
eve
runt
Kr
en
hairy
Nos
ftz
odd


RC
Hyst
Thres
0
10
20
30
40
60
t/s
70
80
90
100
110
120
130
t/s
Cad 1.5 0.75 0
2,5 9,5
Bcd 3.25 0.7 0
5,5
Nos / / 0
gt 2.0 0.4 +1
6 23
47 60
83 96
hb 0.75 0.75 +1
19 43
56 79
Kr 3.25 0.4 0
7,5 16,5
40,5 53,5
76,5 89,5
kni 1.5 0.4 0
0,5 3,5
11,5 21
45 57,5
81,5 94,5
hkb / / 0
eve 2.25 0.5 0
0,5 8
25,5 45
61 81,5
98,5 118
runt 2.5 0.4 +1
17,5 29,5
54 66,5
90 102,5
hairy 4.0 0.4 0
5 13,5
36,5 49
73 85,5
ftz 2.5 0.5 +1
4 8
28,5 39,5
65 76
odd / / 0
en 1.75 0.6 +1
7 11
33 43
70 80



from: J. Krumsiek et al. [18]

Factor	Boolean update rule
GATA-2	$GATA-2 \wedge (\overline{GATA-1 \wedge FOG-1}) \wedge \overline{PU.1}$
GATA-1	$(GATA-1 \vee GATA-2 \vee Fli-1) \wedge \overline{PU.1}$
FOG-1	$GATA-1$
EKLF	$GATA-1 \wedge \overline{Fli-1}$
Fli-1	$GATA-1 \wedge \overline{EKLF}$
SCL	$GATA-1 \wedge \overline{PU.1}$
C/EBPα	$C/EBP\alpha \wedge (\overline{GATA-1 \wedge FOG-1 \wedge SCL})$
PU.1	$(C/EBP\alpha \vee PU.1) \wedge (\overline{GATA-1 \vee GATA-2})$
cJun	$PU.1 \wedge \overline{Gfi-1}$
EgrNab	$(PU.1 \wedge cJun) \wedge \overline{Gfi-1}$
Gfi-1	$C/EBP\alpha \wedge \overline{EgrNab}$

Experiment 41

Simulation of the differentiation of hematopoietic stem cells

In our next experiment, we examine the so-called hematopoietic stem cells [18]. They exist mainly in the bone marrow and are also called blood stem cells.

They are the basis for regeneration of all cells in the blood and the immune system (hematopoiesis); as shown in the figure, a hematopoietic stem cell (HSC) generates four different cell types by activation or inhibition of eleven genes in the associated regulatory network.

Erythrocytes, also called red blood cells, are the most common type of blood cells and are the vertebrates' principal means of delivering oxygen to their body tissues — via blood flow through the circulatory system. They lack a cell nucleus and are dented in the middle. Red blood cells take up oxygen in the lungs or gills and release it into tissues while squeezing through the body's capillaries.

Megakaryocytes (large-nucleus cells) are large bone marrow cells with a multilobed nucleus that are responsible for the production of blood thrombocytes (platelets), which are necessary for normal blood clotting.

Monocytes are the largest white blood cells (leukocytes). They are part of the immune system of vertebrates. They are part of the innate immune system of vertebrates and can destroy foreign microbes and particles by digestion and destruction of this material.

Granulocytes are a category of white blood cells characterized by the presence of granules in their cytoplasm. They are normally found in the bloodstream. Once they have received the appropriate signals, it takes them a short time to leave the bloodstream and reach the site of an infection, where they destroy pathogenic organisms by ingesting them.

The differentiation of these four blood cell types from hematopoietic stem cells is controlled by a biochemical network, as we already know from cell division.

If we use the diagram in the usual way and try to build the circuit, we have our familiar problem again. We do not know either the thresholds of the modules or the weights of the signals. But luckily this time the additional table is very helpful. In it we find all the signals involved with their equivalent Boolean expression from one time step to the next. These are definite, but we must transform them into expressions of threshold logic to be able to construct the network and the setup with the gene modules. At this point, you probably recognize that the diagram shown is just an overview of the activating and inhibiting signals. So let's start the sophisticated work!

A Boolean term like SCL = GATA1 ^ $\overline{\text{PU.1}}$ is simply identifiable as an AND function of both signals **GATA1** and the inverse **PU.1**, and leads to the threshold-function (see exp. 6, page 27) with the representation:

$$\text{SCL} = \langle \text{GATA1} + \overline{\text{PU.1}} \rangle_{2:1}$$

$\overline{\text{PU.1}}$, the needed inverse of PU.1, is unfortunately not available in our system. However, there exists a very important relationship which we can use as a bridge between Boolean and threshold expressions to help us to proceed:

$$\text{PU.1} \vee \overline{\text{PU.1}} = 1$$
$$\text{PU.1} + \overline{\text{PU.1}} = 1$$

This is always true in both realms no matter which variable we take, each variable OR its complement (e.g., inverse) always equal 1. By transposing we get

$$\overline{\text{PU.1}} = 1 - \text{PU.1}$$

We replace $\overline{\text{PU.1}}$ in the threshold equation and now we have with −PU.1, a signal our gene module can

Transformation of the Boolean expressions into threshold expressions

GATA2 := GATA2 ^ ($\overline{\text{GATA ^ FOG1}}$) ^ $\overline{\text{PU.1}}$

:= GATA2 ∧ ($\overline{\text{GATA1}}$ ∨ $\overline{\text{FOG1}}$) ∧ $\overline{\text{PU.1}}$

:= <2GATA2 + 2$\overline{\text{PU.1}}$ + $\overline{\text{GATA1}}$ + $\overline{\text{FOG1}}$>$_{5:4}$

:= <2GATA2 + 2(1 − PU.1) + (1 − GATA1) + (1 − FOG1)>$_{5:4}$

:= <2GATA2 + 2 − 2PU.1 + 1 − GATA1 + 1 − FOG1>$_{5:4}$

:= <2GATA2 − 2 PU.1 −GATA1 − FOG1 + 4>$_{5:4}$

:= <**2**GATA2 − **2**PU.1 − GATA1 − FOG1>$_{1:0}$

GATA1 := (GATA1? GATA2 ? Fli1) ^ $\overline{\text{PU.1}}$

:= <GATA1 + GATA2 + Fli1 + 3$\overline{\text{PU.1}}$>$_{4:3}$

:= <GATA1 + GATA2 +Fli1 + 3(1 − PU.1)>$_{4:3}$

:=<GATA1 + GATA2 + Fli1 + 3 −3PU.1>$_{4:3}$

:=<GATA1 + GATA2 + Fli1 − **3**PU.1>$_{1:0}$

FOG1 := GATA1 (For transformation, see SCL)

:= <GATA1>$_{1:0}$

EKFL := GATA1 ^ $\overline{\text{Fli1}}$ (For transformation, see SCL)

:=<GATA1 - Fli1>$_{1:0}$

Fli1 := GATA1 ^ $\overline{\text{EKFL}}$ (For transformation, see SCL)

:=<GATA1 - EKFL>$_{1:0}$

SCL :=GATA1 ^ $\overline{\text{PU.1}}$ (For transformation, see text)

:=<GATA1 - PU.1>$_{1:0}$

C/EBPα :=C/EBPα ^ ($\overline{\text{GATA1 ^ FOG1 ^ SCL}}$) For transformation, s. text)

:=<3C/EBPα − GATA1 − FOG1 − SCL>$_{1:0}$

PU.1 := (C/EBPα ? PU.1) · ($\overline{\text{GATA1 ? GATA2}}$)

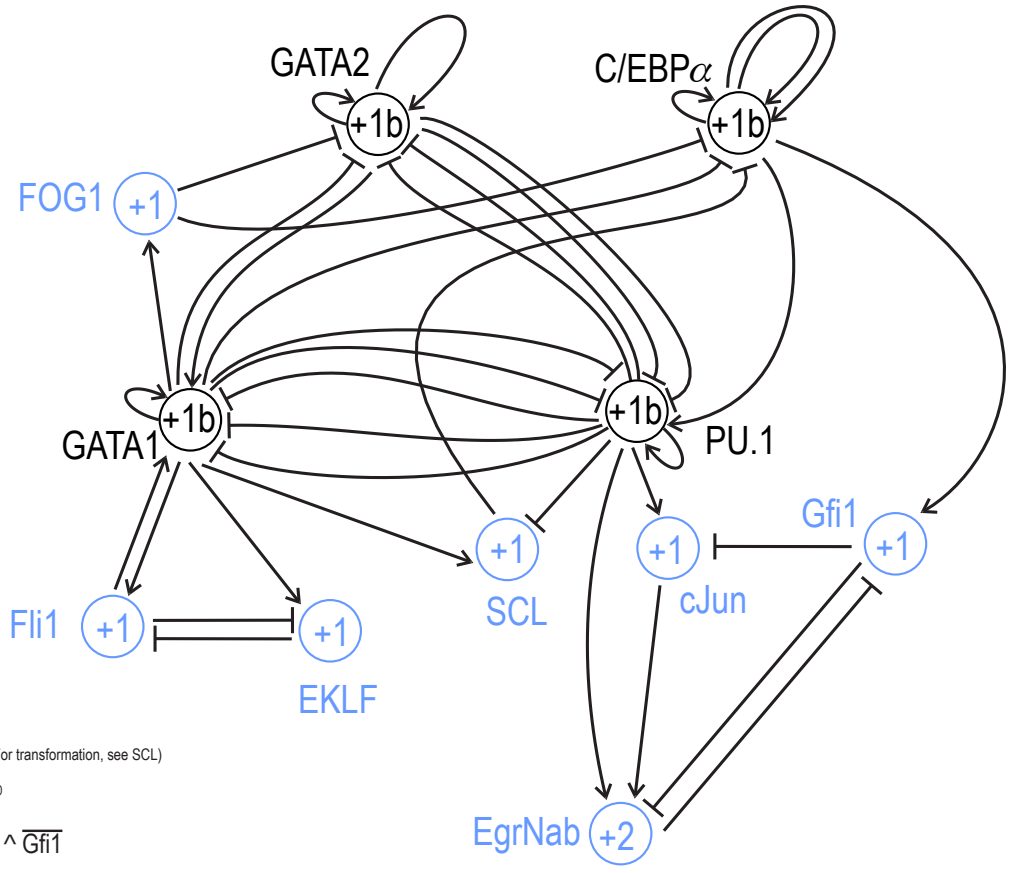
:= (C/EBPα ∨ PU.1) ∧($\overline{\text{GATA1}}$ ∧ $\overline{\text{GATA2}}$)

:= <C/EBPα + PU.1 + 2$\overline{\text{GATA1}}$ + 2$\overline{\text{GATA2}}$>$_{5:4}$

:=<C/EBPα + PU.1 + 2(1 − GATA1) + 2(1 − GATA2)>$_{5:4}$

:=<C/EBPα + PU.1 −2GATA1 − 2GATA2 + 4>$_{5:4}$

:=<C/EBPα + PU.1 − **2**GATA1 − **2**GATA2>$_{1:0}$



cJun :=PU.1 ^ $\overline{\text{Gfi1}}$ (For transformation, see SCL)

:=<PU.1 − Gfi>$_{1:0}$

EgrNab := (PU.1 ^. cJun) ^ $\overline{\text{Gfi1}}$

:= <PU.1 +cJun + $\overline{\text{Gfi1}}$>$_{3:2}$

:= <PU.1 + cJun + (1− Gfi1)>$_{3:2}$

:=<PU.1 + cJun − Gfi1>$_{2:1}$

Gfi1 :=C/EBPα ^ $\overline{\text{EgrNab}}$ (For transformation, see SCL)

:=<C/EBPα − EgrNab>$_{1:0}$

provide.

$$SCL = <GATA1 + 1 - PU.1>_{2:1}$$

For simplifying we subtract the constant 1 in the expression and in the thresholds and finally get something that is realizable with our modules

$$SCL = <GATA1 - PU.1>_{1:0}$$

We verify and find that SCL only gets activated if GATA1 is active and PU.1 is inactive.

As an exercise, here is another more complicated transformation from the table:

$$C/EBPL\alpha = C/EBPL\alpha \wedge \overline{(GATA1 \wedge FOG1 \wedge SCL)}$$

We first convert the Boolean expression

$$C/EBPL\alpha = C/EBPL\alpha \wedge (\overline{GATA1} ? \overline{FOG1} ? \overline{SCL})$$

The »most important« signal is C/EBPLα and it gets the weight 3; the threshold expression for the AND-function between C/EBPLα and the bracketed term with the OR-function of three signals becomes

$$C/EBPL\alpha = <3C/EBPL\alpha + \overline{GATA1} + \overline{FOG1} + \overline{SCL}>_{4:3}$$

The threshold must be +4 so that C/EBPLα will be activated in the next time step if at least one of the inverted signals is active besides C/EBPLα itself.

As in the first example we must now substitute the three inverted signals that are not available:

$$\overline{GATA1} = 1 - GATA1 \text{ etc.}$$

$$C/EBPL\alpha = <3C/EBPL\alpha - GATA1 - FOG1 - SCL + 3>_{4:3}$$

After subtracting the constant 3 in the expression and in the thresholds we finally get:

$$C/EBPL\alpha = <3C/EBPL\alpha - GATA1 - FOG1 - SCL>_{1:0}$$

All other signals in the circuit are transformed in a similar way to threshold terms. The results are listed above with the Boolean expression in the figure. An examination shows that our gene modules luckily have enough inputs so they can get all needed signals. Just as in the last transforming example, the output of **C/EBPLα**? has to give a threefold signal to the same inputs. We can use the »bistable« function of the module. Therefore, the signal has been connected externally only twice to the inputs and there are still inputs available for the three leftover signals.

The connection diagram with all the gene modules and their thresholds is also shown. A possible setup can be found on the next page.

It is readily identifiable in a rough overview that there are several genes forming pairs of antagonists: Therefore either **PU.1** or **GATA1** can be active because they inhibit each other. The same applies to **PU.1** and **GATA2**, **Gfi1** and **EgrNab**, and **EKLF** and **Fli1**.

Before we apply the current supply, we verify that the power supply is not connected to an output and that two outputs do not work directly against each other.

RC and hysteresis at all modules are in the middle position. After applying the power supply, nothing happens. This model is similar to that for Arabidopsis; we have to start it manually. The two models resemble each other in another way, too: The activities of the Arabidopsis genes developed four different flower leaves; in this model, four different blood cell types develop (see figure on page 137).

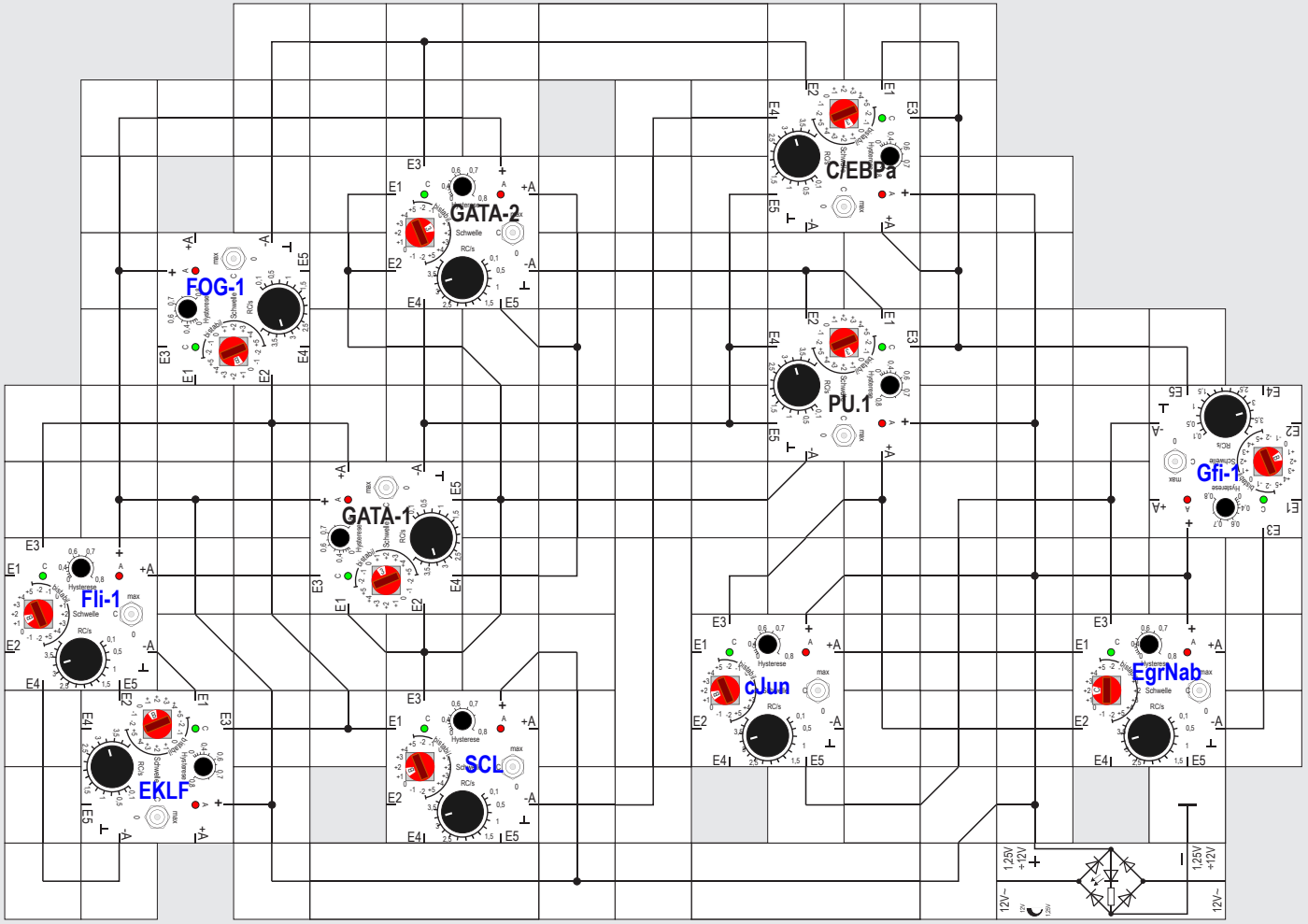
The best way to start is by turning the toggle switch for the **C/EBPLα**-gene module briefly from »C« to »max« and back again. Consequently, **PU.1** and **Gfi1** become active and remain in this state. We are now in the right branch of the diagram, so granulocytes would be produced.

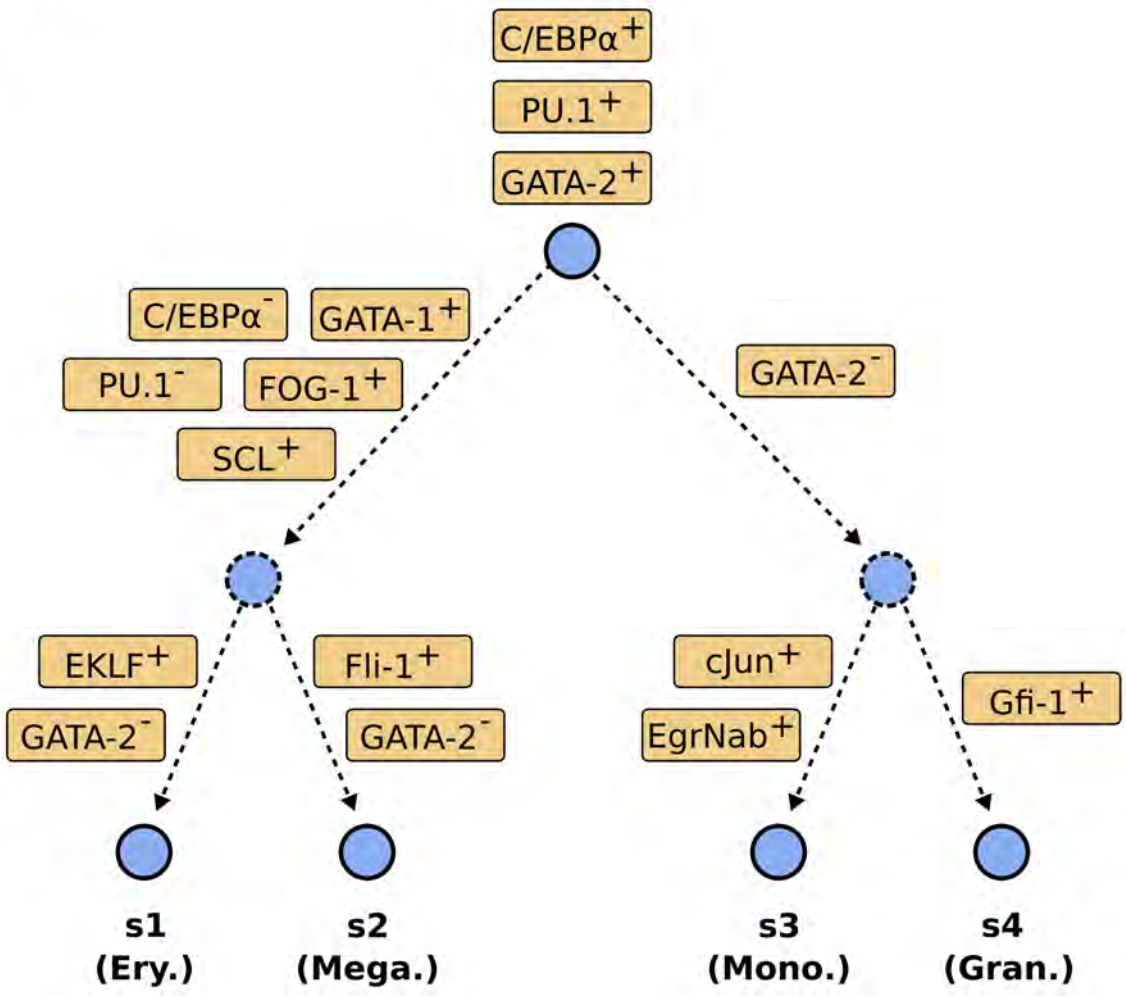
If we activate **EgrNab** with its toggle switch (turn it a little bit longer on »max« before turning it back to »C«), **Gfi1** becomes inactive; however, **EgrNab** and also **cJun** remain active. Monocytes are produced.

To get to the other two branches of the diagram, we activate **GATA2** manually (moving it briefly to »max« and back again).

We can observe that first **PU.1** becomes inactive, followed by **cJun** and **EgrNab**. Simultaneously, **GATA1**, **SCL**, **FOG1**, **EKFL**, **Fli1** (and for a moment **Gfi1** again) become active. **SCL** finally turns off **C/EBPLα**? and therefore **Gfi1**, too. **GATA2** doesn't remain active, either.

Since **EKLF** and **Fli1** are antagonists, the precise

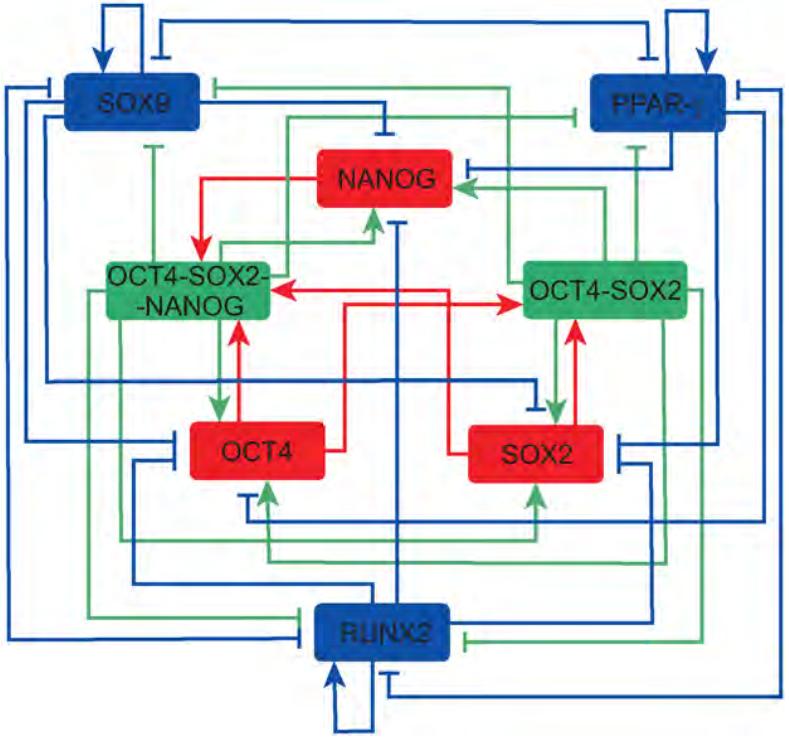

C/EBPa
GATA-2
FOG-1
PU.1
Gfi-1
GATA-1
Fli-1
cJun
EgrNab
SCL
EKLF



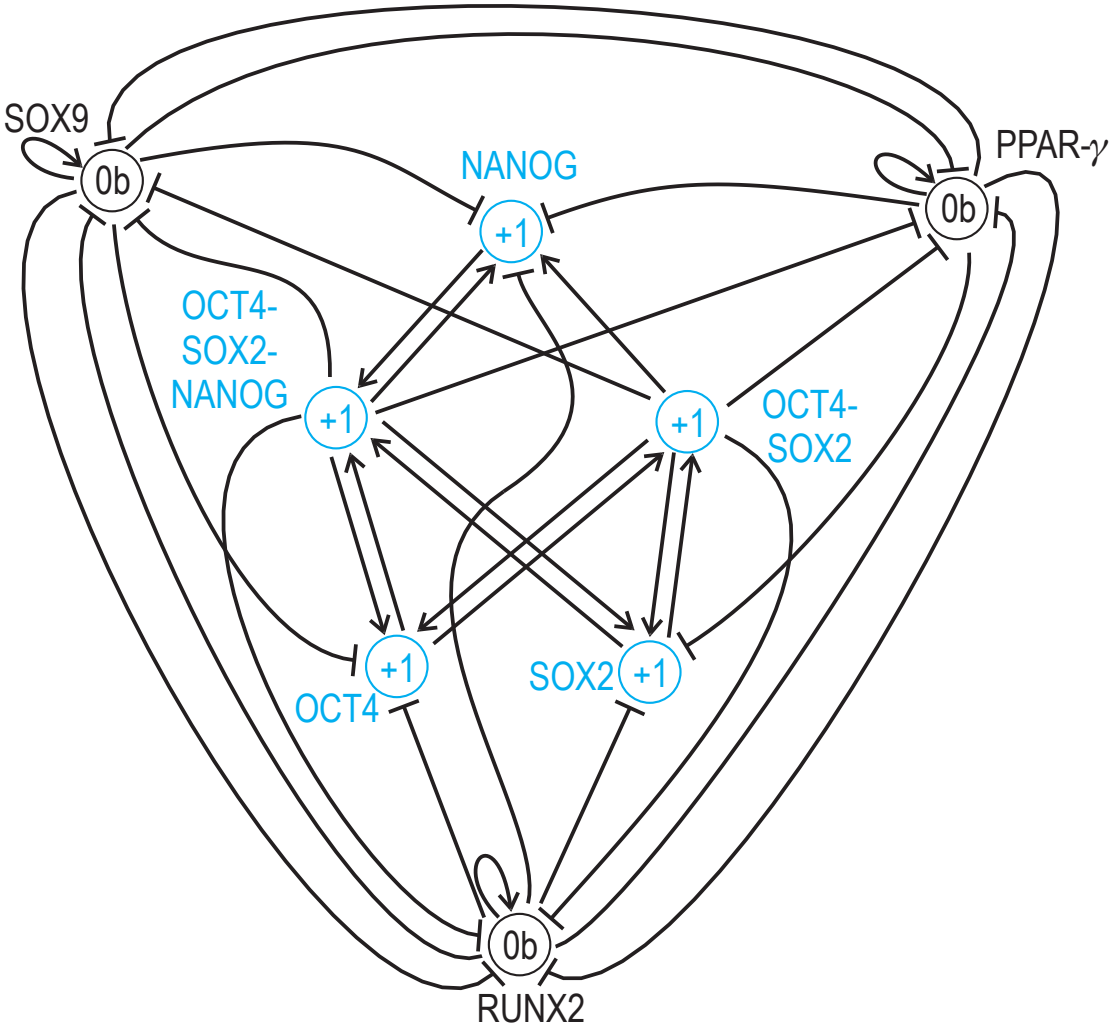
time adjustment and the hysteresis are crucial for which module becomes first active and inhibits the other. If it is **Fli1**, megakaryocytes will be produced; if **EKLF** is first, erythrocytes will be produced. We can change between them by turning the active module off momentarily or by turning on the inactive module.

The starting state with all genes inactive appears to be best if we manually deactivate GATA1 for a moment. All genes in the left branch of the circuit turn off then.

We set aside an extension of the circuit with a periodic restart because this would need many extra connections, like the Arabidopsis setup.

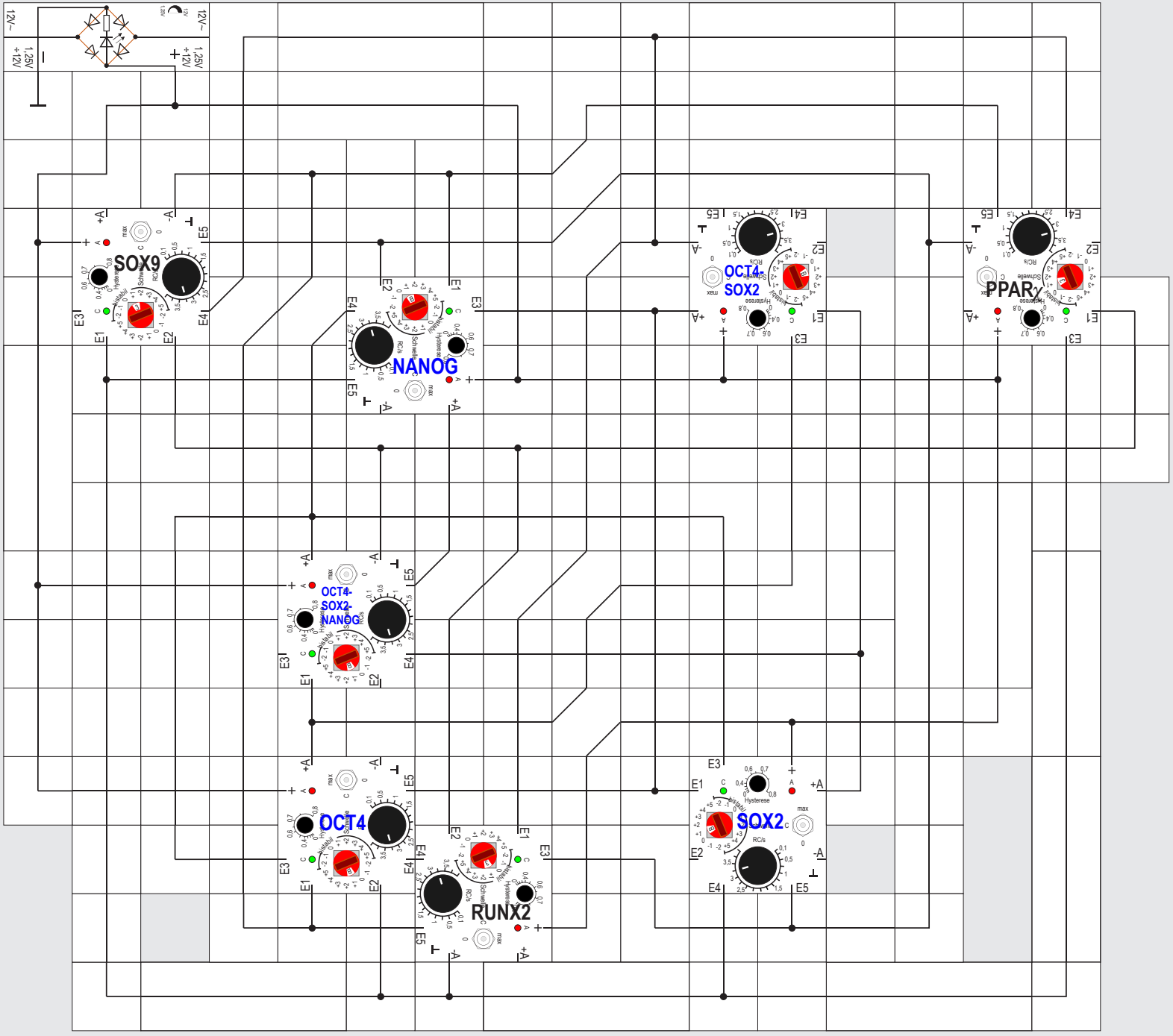


From: MacArthur et al. [19]



Experiment 42
Differentiation of stem cells

We want to end our simulations of biological gene regulation networks with an experiment on a highly topical subject, the differentiation of stem cells. Stem cells have the ability to divide themselves in cell cultures arbitrarily often and then develop to more specialized cells. This is easy to explain by looking at the development of the human embryo: When a sperm fertilizes an egg cell, at first a single cell with the potential to form a complete organism results. This fertilized egg cell is totipotent, meaning its potential is total because it can differentiate into every other kind of cell. This cell divides into several identical cells within the next hours that are totipotent, too. If they had been implanted in a womb separately, a fetus would have grown from each of them. Monozygotic twins that are genetically identical emerge like this. After about four days, specialization starts, and two sets of cells develop; an external mass of cells forming a hollow shell, and an internal mass called the blastocyst.
This internal mass of cells will develop into the different body tissues of the fetus, whereas the external mass of cells form structures including the placenta, which supplies the fetus with nutrients during development.
The internal mass of cells can differentiate to every conceivable cell in the human body, but it is no longer able to develop an organism because it can't form a placenta anymore. These cells' potential is no longer total but only pluripotent. These pluripotent cells can form into the three germ layers of the embryo, the inner, outer, and middle layers that further specialize into various organs and tissues.
Pluripotent stem cells specialize further to stem cells, from which certain cell lines can develop with a specific function. For instance, blood stem cells develop into red and white blood cells and thrombocytes as described in the previous experiment. These even more specialized stem cells are called multipotent.
It should be obvious that understanding the processes of cell division and differentiation into certain tissues or stem cells is of great interest for medical progress. At the center of the research are the factors that determine when and in which way certain genes are activated or inhibited. All this interest is driven by the hope of obtaining a better understanding of diseases based on malfunctions in gene regulation, such as cancer. An additional hope is that cell tissues could be created from pluripotent stem cells to repair or replace damaged organs.
The gene regulation network chosen for our experiment is responsible for differentiation to endoderm, ectoderm and mesoderm — the inner, outer, and middle germ cell layers. It shows how differentiation from pluripotent blastocyst cell clusters is managed by the control of three genes, **OCT4**, **NANOG** and **SOX2** (marked red in the figure) [19]. These three genes activate each other indirectly via two other genes marked in green and together form the inner core of the network. They also inhibit via the two green genes the »outer« genes marked in blue, **SOX9**, **PPAR-γ** and **RUNX2**. These each inhibit the two others and the three genes of the core, **OCT4**, **NANOG** and **SOX2**, directly. The mutual inhibition of **SOX9**, **PPAR-γ** and **RUNX2** is different from what we saw for the repressilator. There, one of the three genes involved inhibited only the following one, so an oscillation occurred. Here, one gene inhibits **all** the others; once it is active, it doesn't allow any other gene to be active anymore. We already met this type of network in experiment 20 in its pure form. So direct inhibition is very strong. This is how stabilization of a cell type should be! This is exactly why the different cells of our body keep their identity lifelong. Recent research has shown quite spectacularly that differentiated cells can be reprogrammed by use of a trick! Here, a little cocktail of a few proteins is injected into a cell in the laboratory, which becomes a pluripotent stem cell after


12V~
1,25V
+12V
SOX9
NANOG
OCT4-
SOX2-
NANOG
OCT4
RUNX2
OCT4-
SOX2
SOX2
PPARγ

a while. This was once thought impossible, and thus John B. Gurdon and Shinya Yamanaka were awarded the Nobel Prize in Medicine in 2012 for demonstrating that mature cells can indeed be reprogrammed to a pluripotent cell state. This approach has even succeeded in creating stem cells from typical skin cells of a mouse, thereby producing a clone of the mouse! Turning back to our example that we want to set up next, sadly, there are again no thresholds given in the illustration of the regulation network. A short reflection will show that they can be at maximum +1 for the core and 0 for the outer modules (whether bistable or not doesn't matter) if we want that anything to happen when we turn on the supply voltage. If we raise the thresholds of the outer ones to +1, everything would remain inactive and it would only be possible to activate one module manually with the switch, which would inhibit the other ones. Then those modules could not even be activated manually. Transposed into biological processes, this means that once it is determined which germ layer — inner, outer or middle — the cell will develop into, its fate cannot be reversed.

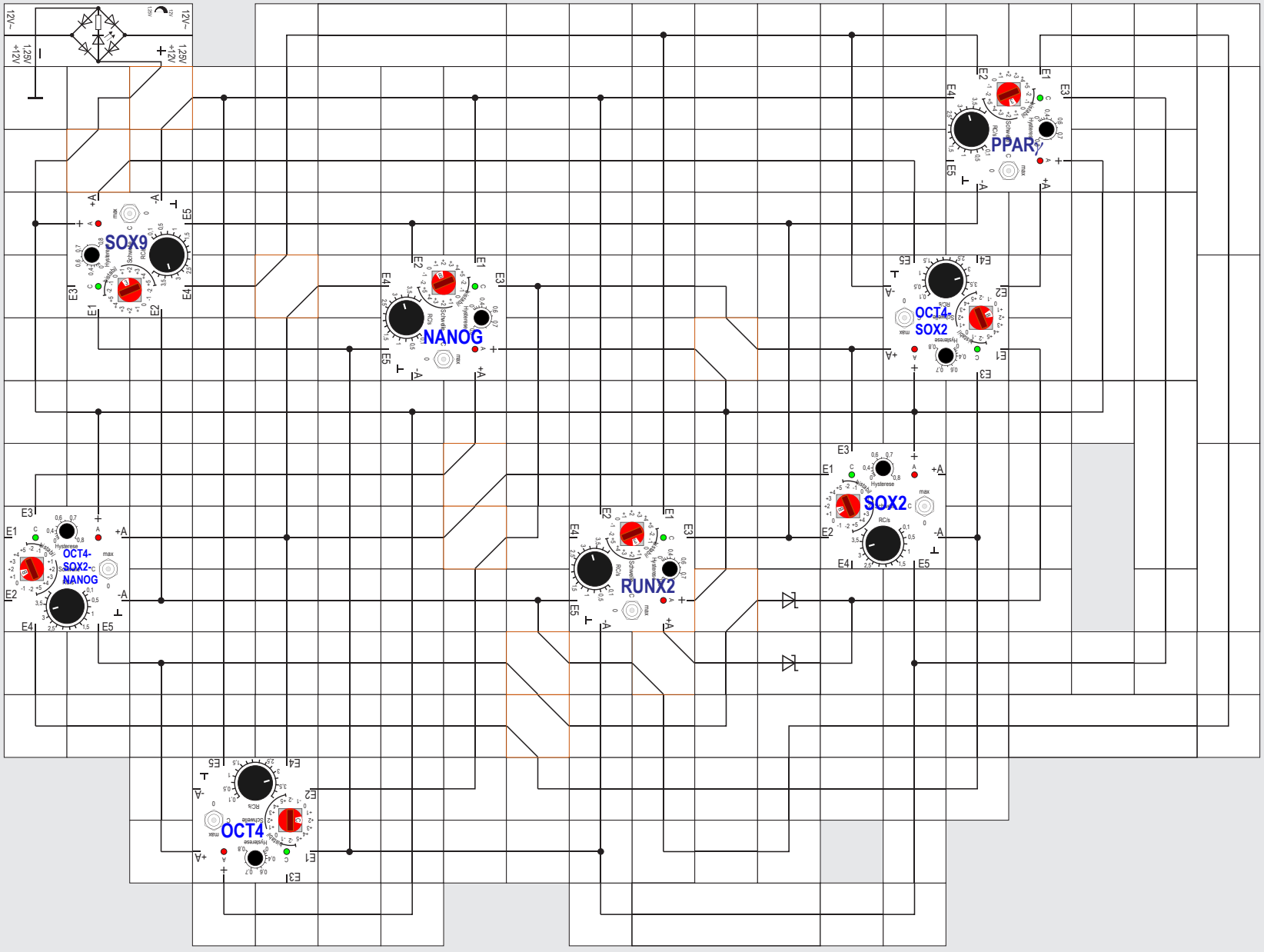
On page 138 is a gene network on the left as displayed from MacArthur et al. [19] and on the right the corresponding connection diagram we are used to, including thresholds. For **SOX9**, **PPAR-γ** and **RUNX2**, the thresholds are 0. Based on this figure, a possible setup would be like the figure on the previous page. The RC times of the three mentioned gene modules have the same values, so after first applying the supply voltage, without manual intervention, at maximum all three of them will be activated. But it does not remain like this: Depending on the tolerance, one of the genes remains active and inhibits the others, which have possibly been active for a moment. This is everything that happens.

Only by manual activation of one of the other genes (preferably **OCT4-SOX2** or **OCT4-SOX2-NANOG**) will the activated outer gene be inactivated and, one after another, all inner genes be activated. This corresponds to a further differentiation in biological reality that cannot be undone. The mutual activation of these now turned-on genes is so strong that it can only be deactivated by either manual activation of all outer genes or by manual deactivation of **OCT4S-OX2** and **OCT4-SOX2-NANOG**. If we then switch all outer genes back to the »C« position again as in the beginning, only one of them remains active.

In our model, the activated inner genes represent differentiated cells. Now, what about the Nobel prize-winning induced pluripotent stem cells of Yamanaka? Let's consider how it might be possible to undo the dynamics of differentiation. Reprogramming in laboratories indeed uses the essential proteins such as **OCT4**, **SOX2** and NANOG or related proteins in their cocktails, for a reprogramming similar to our experiment here. Which combination works best is the topic of this very active research into pluripotent stem cells.

Let's conclude this experiment by attaching a setup to show the effect when differentiation and redifferentiation alternate automatically. By trial and error with the model, you can find out that the inner gene modules can be activated by an activating signal (weight 1) at one of the three free inputs of **OCT4-SOX2**. This enables an elegant toggling between the active inner and active outer genes for demonstration purposes. But it has to be emphasized that this is exactly what does not happen in reality, because otherwise the development of the organism won't proceed properly.

OCT4-SOX2 and subsequently the whole inner network can be turned on by an activating +A signal of either **SOX9**, **PPAR-γ** or **RUNX2**; one of these genes is always active and the **OCT4-SOX2** module has three free inputs left (an OR operation would also be possible if the number of inputs were insufficient). If the inner network is active, the activating outer gene will be deactivated.

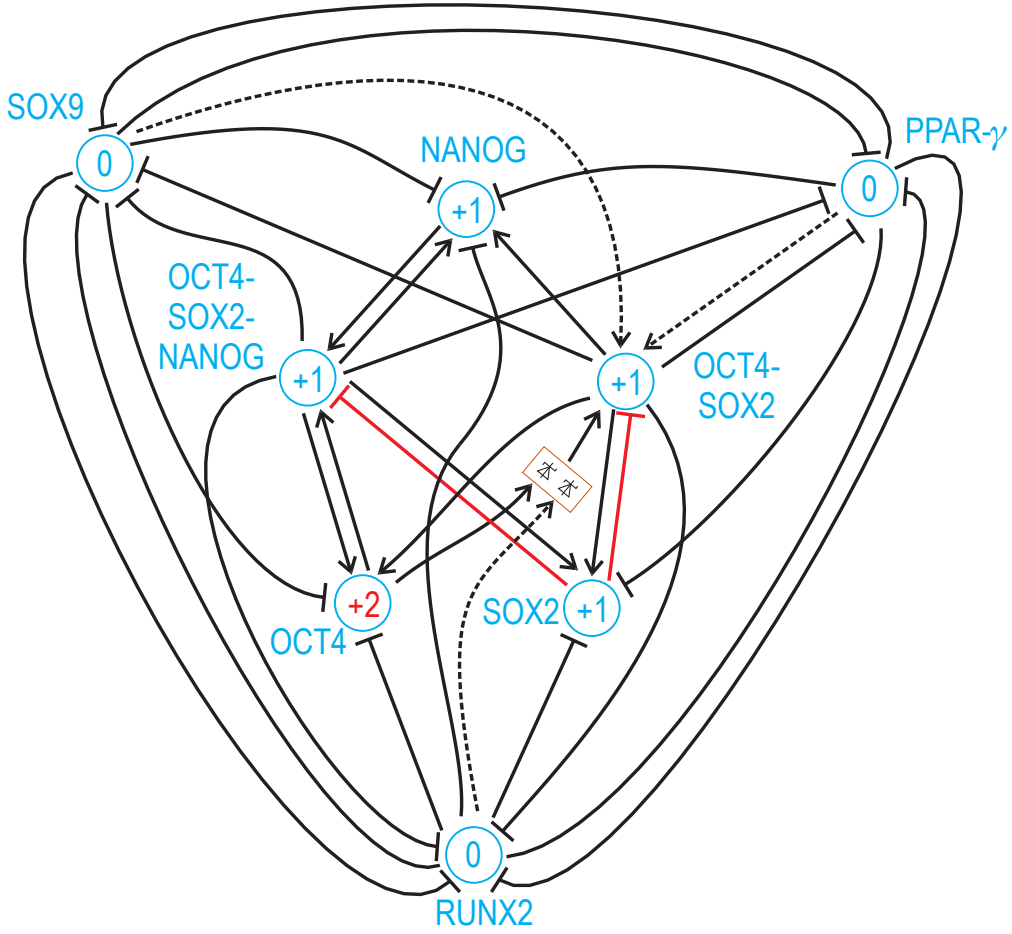

12V~
125V
+12V
SOX9
NANOG
PPARγ
OCT4
SOX2
OCT4-
SOX2-
NANOG
SOX2
RUNX2
OCT4

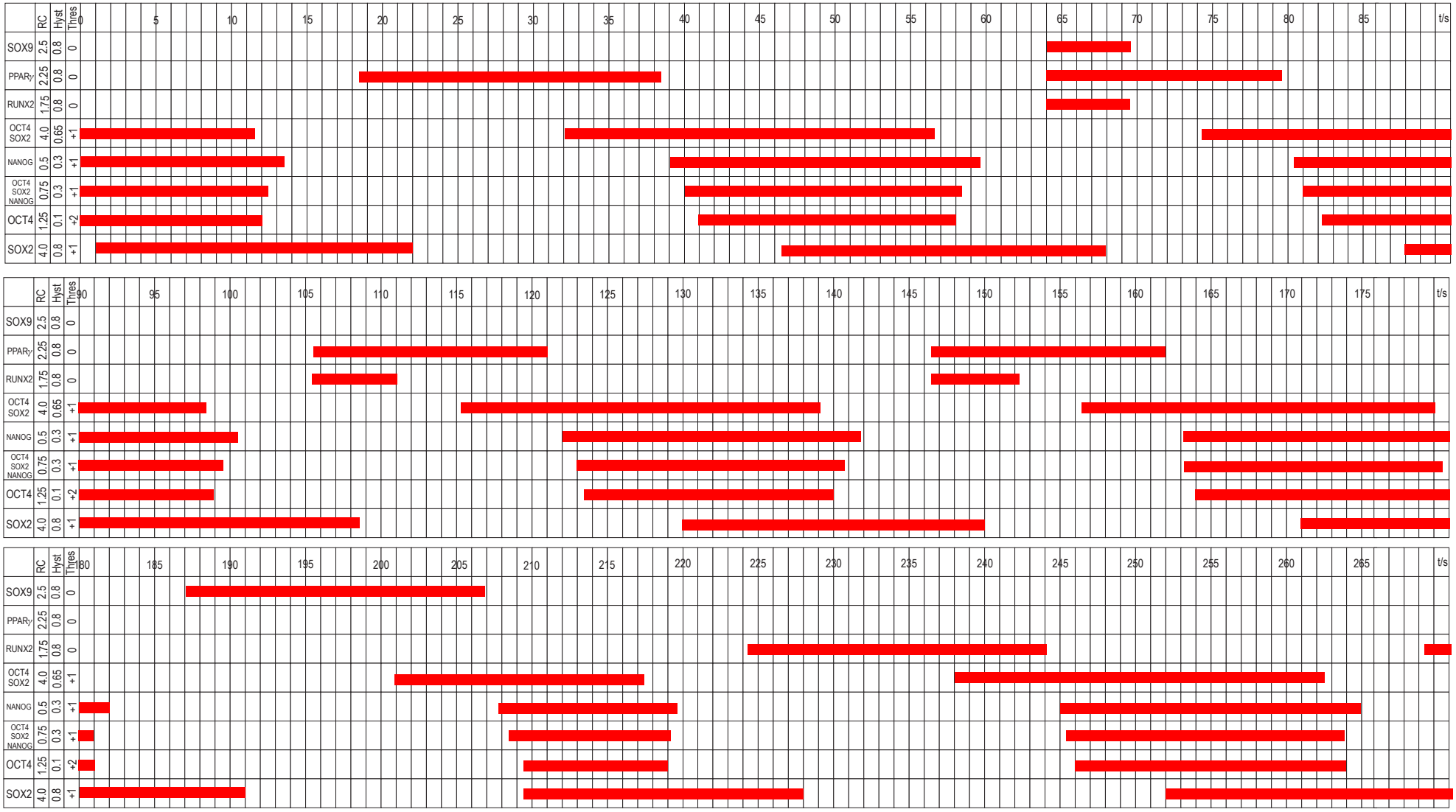
Backspacing, meaning turning off the inner network, is most easily done by a large RC value of **SOX2**; **SOX2** will then be the last active gene, and its inhibiting -A signal can deactivate all inner genes when connected to **OCT4-SOX2-NANOG** and **OCT4-SOX2**, so at least one of the outer genes turns on. But we have to intervene heavily: The inhibiting **SOX2** signal has to be connected to both gene modules with weight 2 to have any effect.

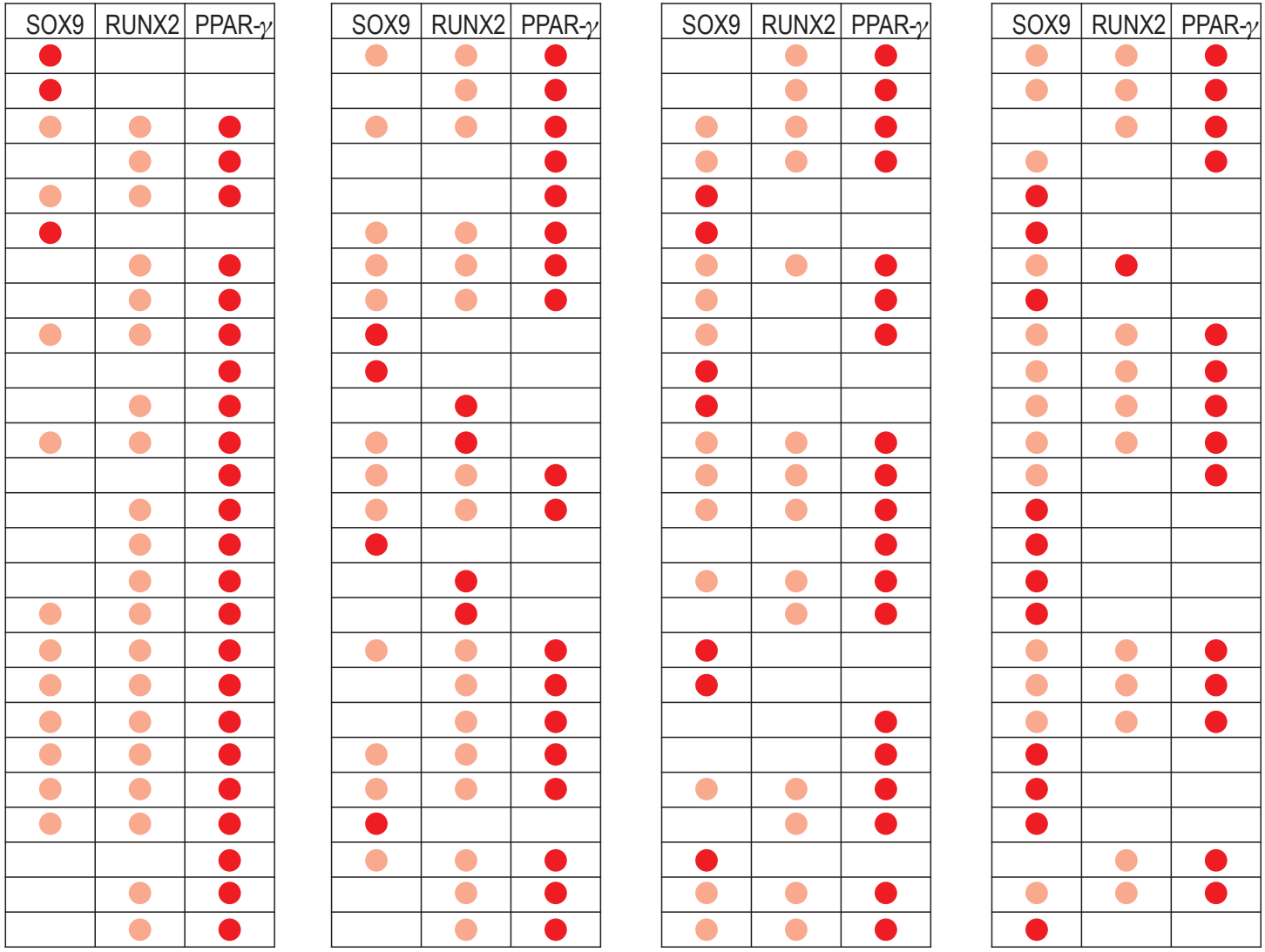
Simplifications in the setup result from the fact that at first **SOX2** sends an activating signal to both **OCT4-SOX2** and **OCT4-SOX2-NANOG** (weight 1), but now an inhibiting signal of weight 2 is added, so as a sum, an inhibiting signal with weight 1 is left; plus the threshold of **OCT4** has to be increased to +2.

The changed connection diagram is displayed on the right. On the previous page is a possible setup in which, for cost reasons, the +A signals of **RUNX2** and **OKT4** are connected in an OR operation.

The following page shows the time course of this setup. In 270 seconds, almost seven runs are displayed, with every run lasting between 41 and 45 seconds, depending on what outer genes have been activated. It is interesting to see that not only is one outer gene always active, but temporarily two or sometimes three, until one of them prevails and




RC
Hyst
Thres
SOX9 2.5 0.8 0
PPARγ 2.25 0.8 0
RUNX2 1.75 0.8 0
OCT4 SOX2 4.0 0.65 +1
NANOG 0.5 0.3 +1
OCT4 SOX2 NANOG 0.75 0.3 +1
OCT4 1.25 0.1 +2
SOX2 4.0 0.8 +1
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 t/s
90 95 100 105 110 115 120 125 130 135 140 145 150 155 160 165 170 175 t/s
180 185 190 195 200 205 210 215 220 225 230 235 240 245 250 255 260 265 t/s



In the next image, showing 104 successive runs, we see that this is usually **PPAR-γ** (shown with a red circle), probably because of inevitable tolerances, but sometimes **SOX9** or **RUNX2** prevail, too. The pink circles show which genes are also activated at first before one of them, namely the red one, dominates. No periodic pattern can be observed. Isn't this amazing? Up to now, we were only concerned with very reliable networks like the cell cycle, where a periodic pattern blinks like clockwork. In developmental biology, situations occur by chance. Before the cell differentiates and finds its final identity, signals from outside often have some influence on what kind of cell type it will become — for instance, the signals of surrounding cells inside a multicellular organism, or due to chance, when several cell types are required at the same time. Our blinking network for demonstration purposes gives a taste of the effects of chance in cell differentiation and scratches the surface of chaos theory's phenomena.

Your own experiments

We have now arrived at the end of our small excursion to biological regulation networks. Now it's your turn! To those who want to build more simulations about gene regulatory networks but don't have examples, it isn't a bad idea to check the Internet. Under the keyword »gene regulatory networks« or something similar in a search engine, you will certainly find some other interesting ones. If you search for images, you will see several examples of regulatory networks, and it is obvious at first glance whether the complexity of these networks is worth a setup. Some are too complex, so you will lack modules and space to build them; some are too easy.

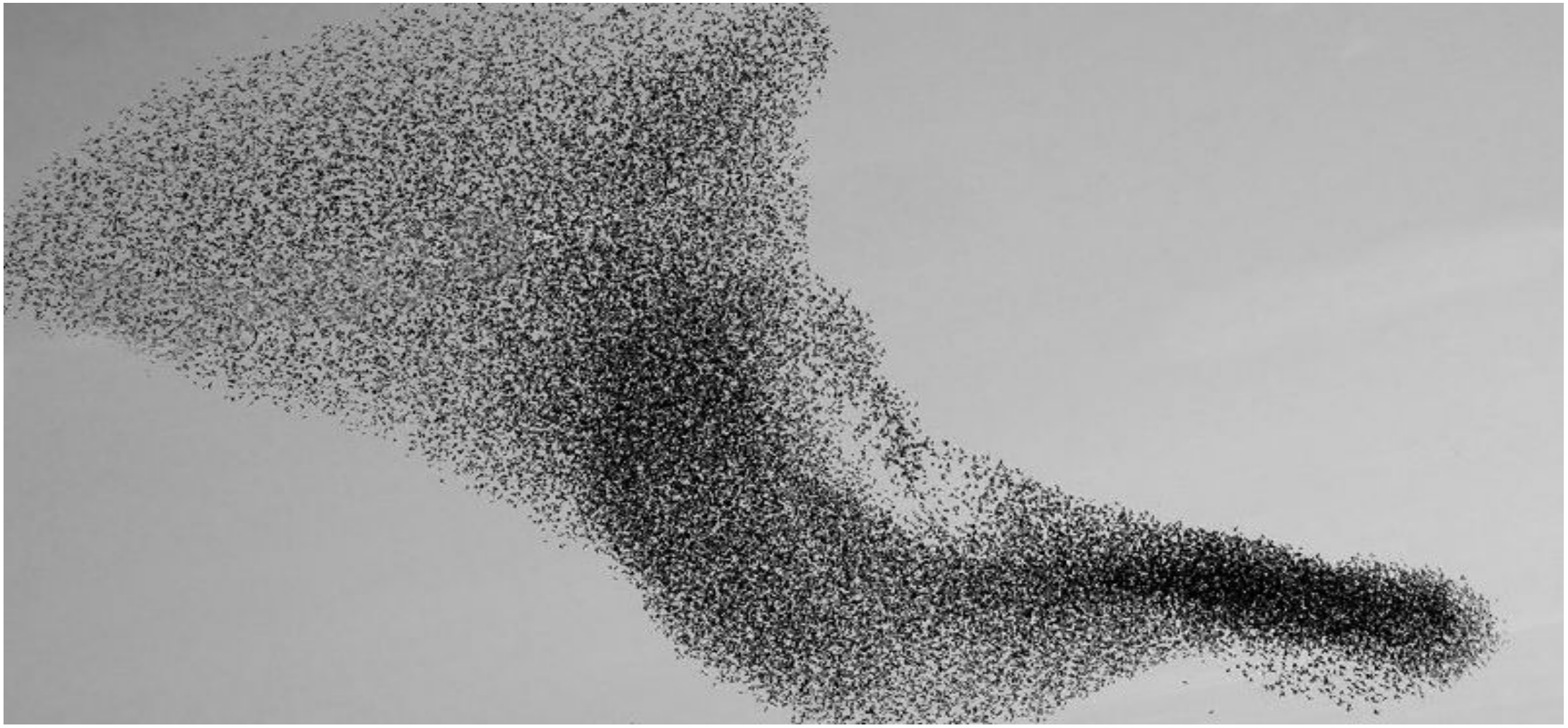
Will you join us?

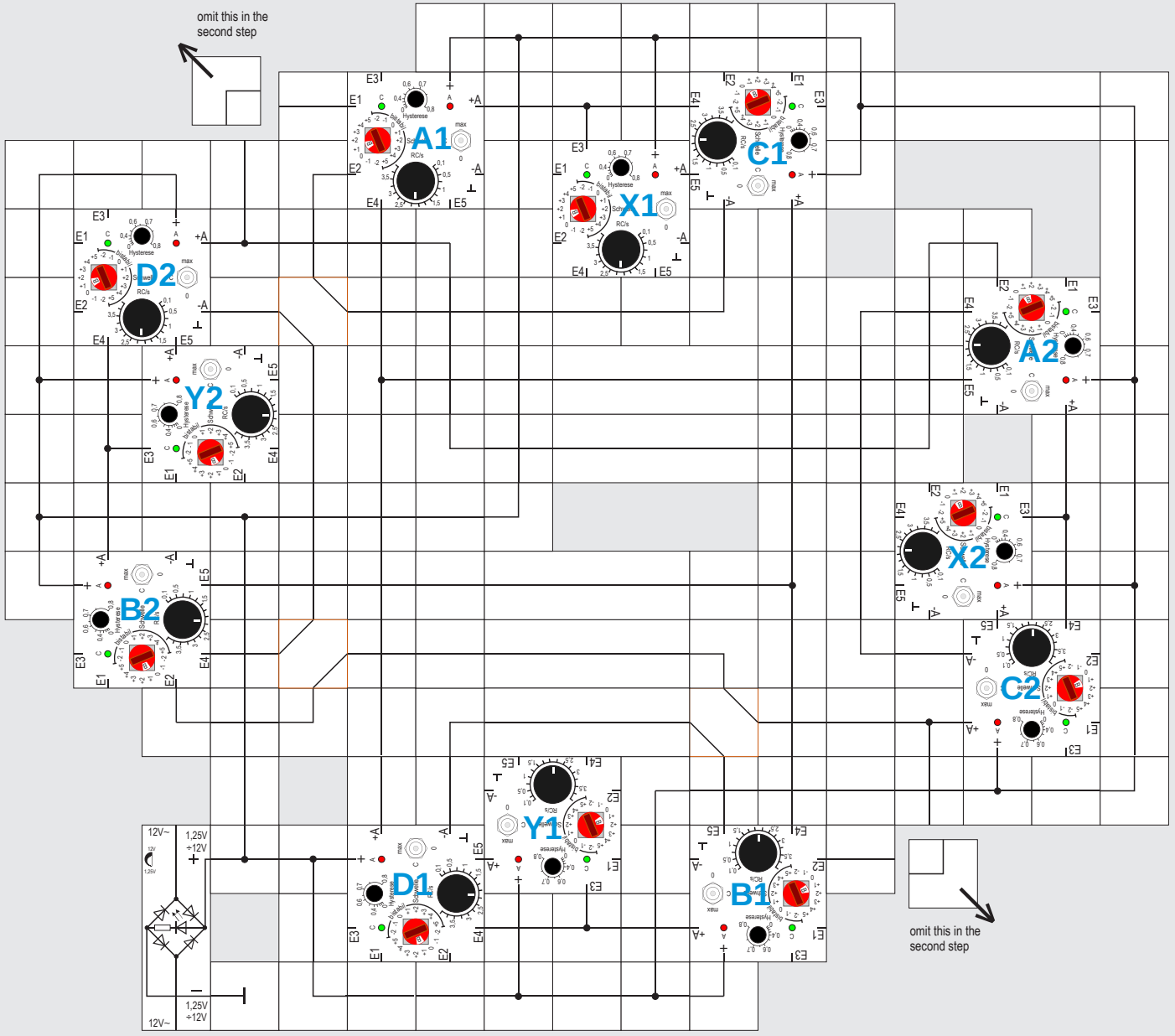
The question is: How will you go on? You have become familiar with all aspects of the LECTRON gene module; you know how it works and how in particular different tasks can be solved with circuits of several gene modules. You are an expert now. We are sure you will now try your own circuits.

Maybe very different networks are possible: Are you able to simulate neuron networks? Or games and logical puzzles? The only limit is your imagination! Do you want to tell us what you've invented? Take a photo with your cell phone and don't hesitate to send it with a short explanation to the authors at:

bornholdt@itp.uni-bremen.de

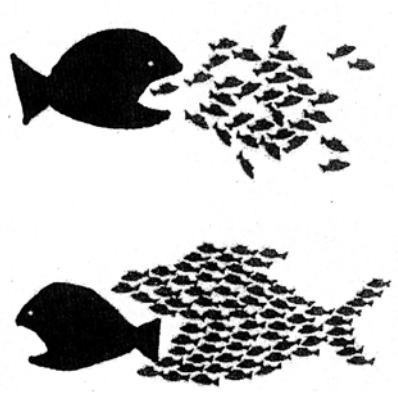
Maybe we can put it into the next edition of this handbook. Of course, LECTRON reserves the right to decide which of the programs sent in will be published. But everyone whose program is printed will receive a volume of the new, expanded edition of the LECTRON experimental handbook for free.




omit this in the second step
A1
X1
C1
D2
A2
Y2
X2
B2
C2
Y1
D1
B1
12V~
1,25V ÷12V
omit this in the second step

5. Swarm behavior of microbes

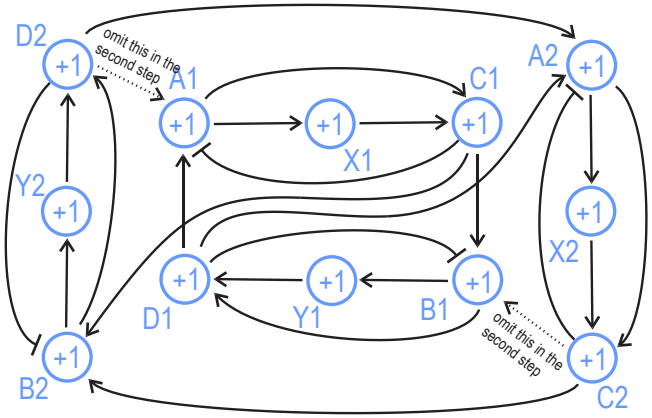
After having learned about a number of examples of gene regulation of living cells and organisms in the previous experiments, we now widen our horizons to ecological phenomena beyond single organisms for the last biological part of the experimental kit. A particularly interesting dynamic phenomenon of communities of organisms is the forming of swarms, or more generally, the coordinate behavior of a group of organisms to gain a common advantage. We will get to know more about this phenomenon with simple models for forming swarms or coordinating microbes. Swarms can be observed not only in groups of fish and birds, but also for smaller life forms like bacteria. Obviously this has some survival advantages compared with a disordered number of individuals, for example when predators get confused by the synchronized behavior of a large number of prey. This is expressed very well in the figure.



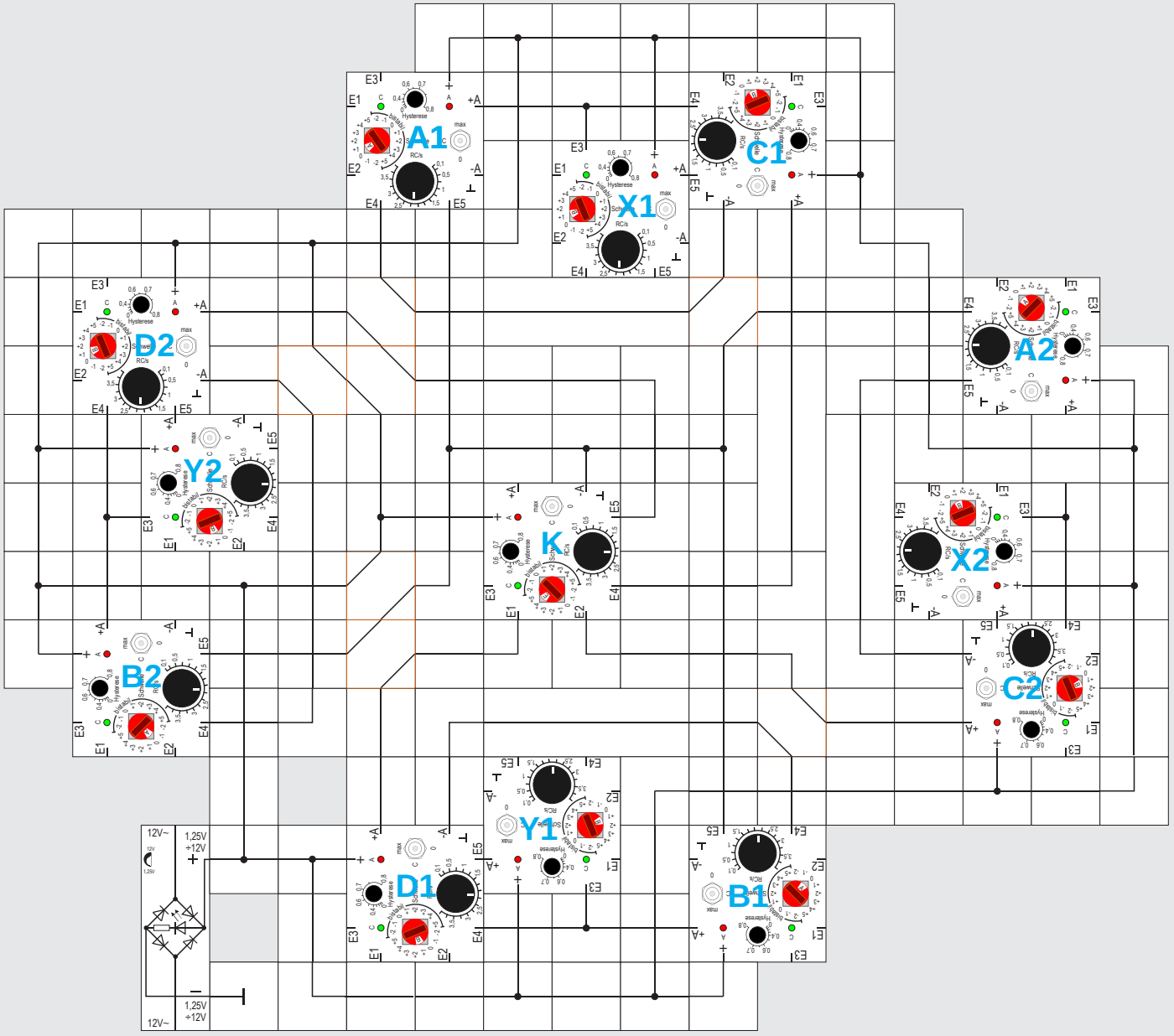
Experiment 43
Forming of swarms

Often swarms are formed when the environmental conditions deteriorate, so many individuals come together, and this bond can increase their chances of survival by differentiation of specialized substructures. For this to happen, an organized exchange of information is inevitable, and we can expect this to be predetermined in gene regulation [11].

Our initial point is a pair of coupled oscillators, as we have already built in experiment 26. So we build a second pair of oscillators and try to find an appropriate coupling (see figure). The assumption is that when a gene produces a protein, it may not only affect its own cell or organism, but also affect cells or organism in its environment, and thus serve as a signaling protein.



We connect the outputs of our first pair with the inputs of the second and vice versa: C1→B2, D1→A2, C2→B1, D2→A1. After applying the supply voltage, we have to start manually, but after a short time we find that all genes are active and don't oscillate anymore. Increasing the thresholds of all input modules to +2 results in the opposite, and the oscillation dies as well. But if we couple a bit less, only C1→B2 and D2→ A1 (remove two angle connections from the setup), we get the desired behavior. Both pairs oscillate synchronously with C1 and C2 in phase and almost in paraphase to D1 and D2. Again, as in experiment 26, some kind of drifting can be observed, but


A1
C1
X1
D2
A2
Y2
K
X2
B2
C2
Y1
D1
B1
12V~
1,25V
÷12V
12V~

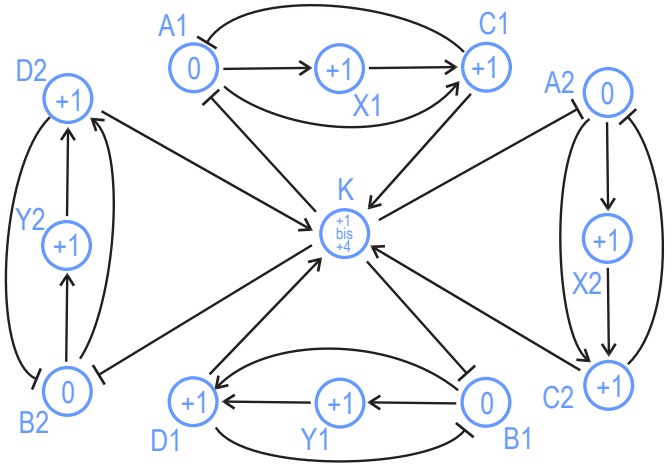
Experiment 44
Quorum sensing

We will get an impressive image of swarm formation if we remove the phase opposition of the synchronous oscillators in our setup. Thus we return to the initial structure of our oscillator and build four of them. First, we try to adjust all frequencies to the same value. When the supply voltage is applied to the setup (first without module K or with module K shut down), the modules should oscillate as long as possible in phase, but independently. However, prior experiments showed that they will drift apart in the long run.

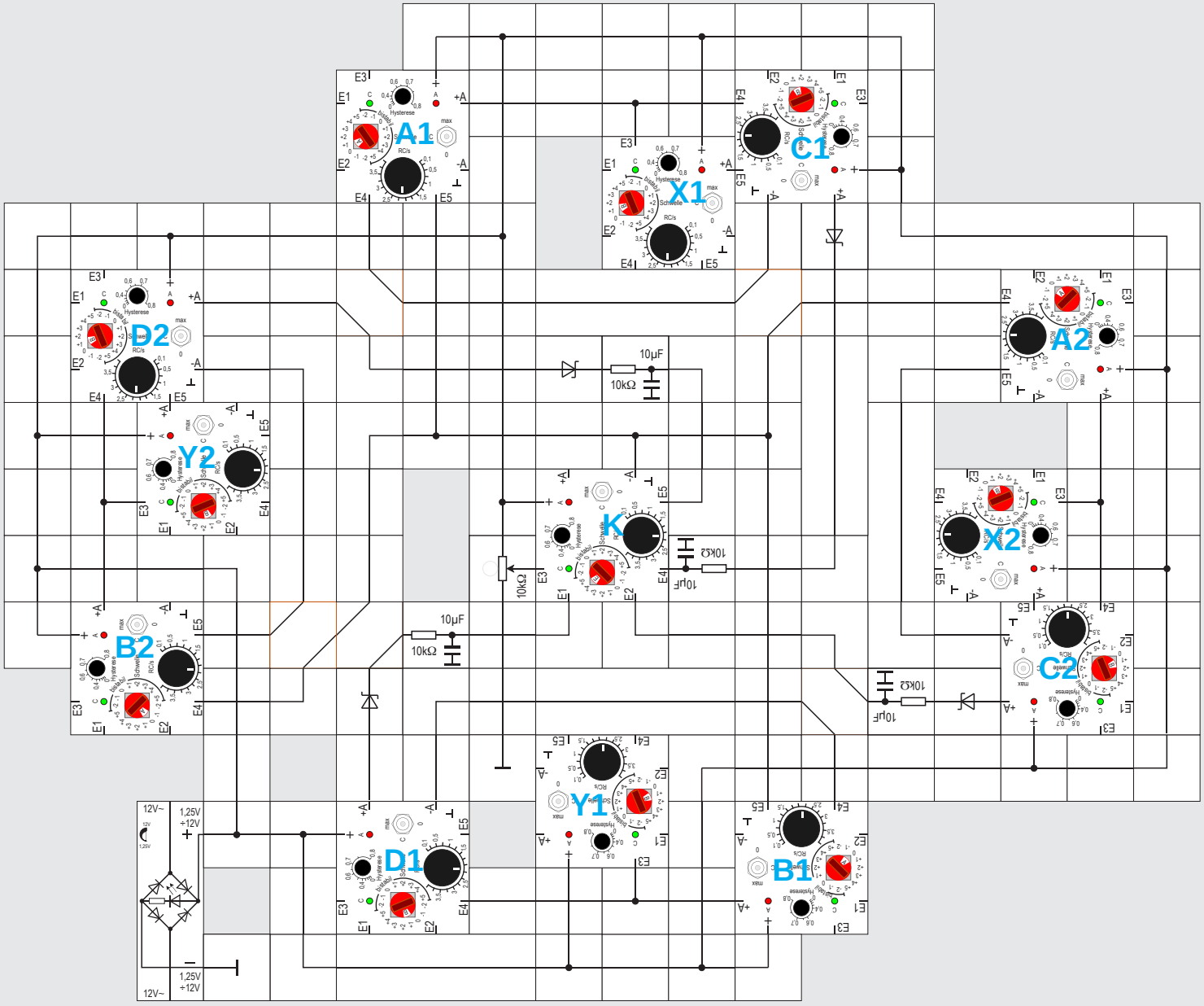
The coupling that prevents this and which we want to implement now is based on the observation in nature that an active cell or organism (here our oscillator) produces a messenger molecule (called an autoinducer), which doesn't only affect itself but also, by diffusion, other similar structures in its immediate surroundings. Since this applies for the other structures, too, they influence each other. When a certain concentration of a substance is reached in the common environment, which can be considered a fluid, genes will be activated that produce the chemical messenger (the autoinducer) so that feedback starts. A well-known example of a protein driven by an autoinducer in bacteria is the enzyme *luciferase*, which is responsible for bioluminescence, as in fireflies. In our experiment, our oscillator will produce the autoinducer.

To model this fluid, our gene modules are perfect because they contain a summing unit and a comparator. The additional module K, which represents the surrounding fluid, therefore gets the signal +A from every oscillator. It adds them, and we can determine when it actively affects the oscillators by the threshold, hysteresis and RC adjustments. In our model, this happens by use of inhibiting connections: The active phase of the modules A1, A2, B1, and B2 can be shortened by the -A output signal of K. This ability for synchronization can be observed in unicellular organisms like *E. coli* and is called »quorum sensing«. It allows the cells to activate genes only to coordinate processes that would be inefficient if they were done only by single cells.

If we switch K to the »0« position, the oscillators vibrate independently and their phase drifts apart.



By adjustments of threshold and time of K, we can choose how strong the mutual coupling should be (and then switch K to the »C« position). Short times and a low threshold of +1 lead to fast synchronization, whereas with threshold +4 and middle to long times, it feels like an eternity until all of them oscillate synchronously. This is because all of the signals have to be active at the same time for a sufficiently long time for the module to be able to send a synchronization signal. We can literally see how hard K tries to create synchronization by watching the brightness of the green LED. But it can never succeed if its time is longer than the oscillator times.


A1
X1
C1
D2
A2
10µF
10kΩ
Y2
K
10kΩ
10µF
X2
B2
10µF
10kΩ
C2
10kΩ
10µF
Y1
D1
B1
12V~
1,25V
÷12V
1,25V
÷12V
12V~

Experiment 45
Improved model I for quorum sensing

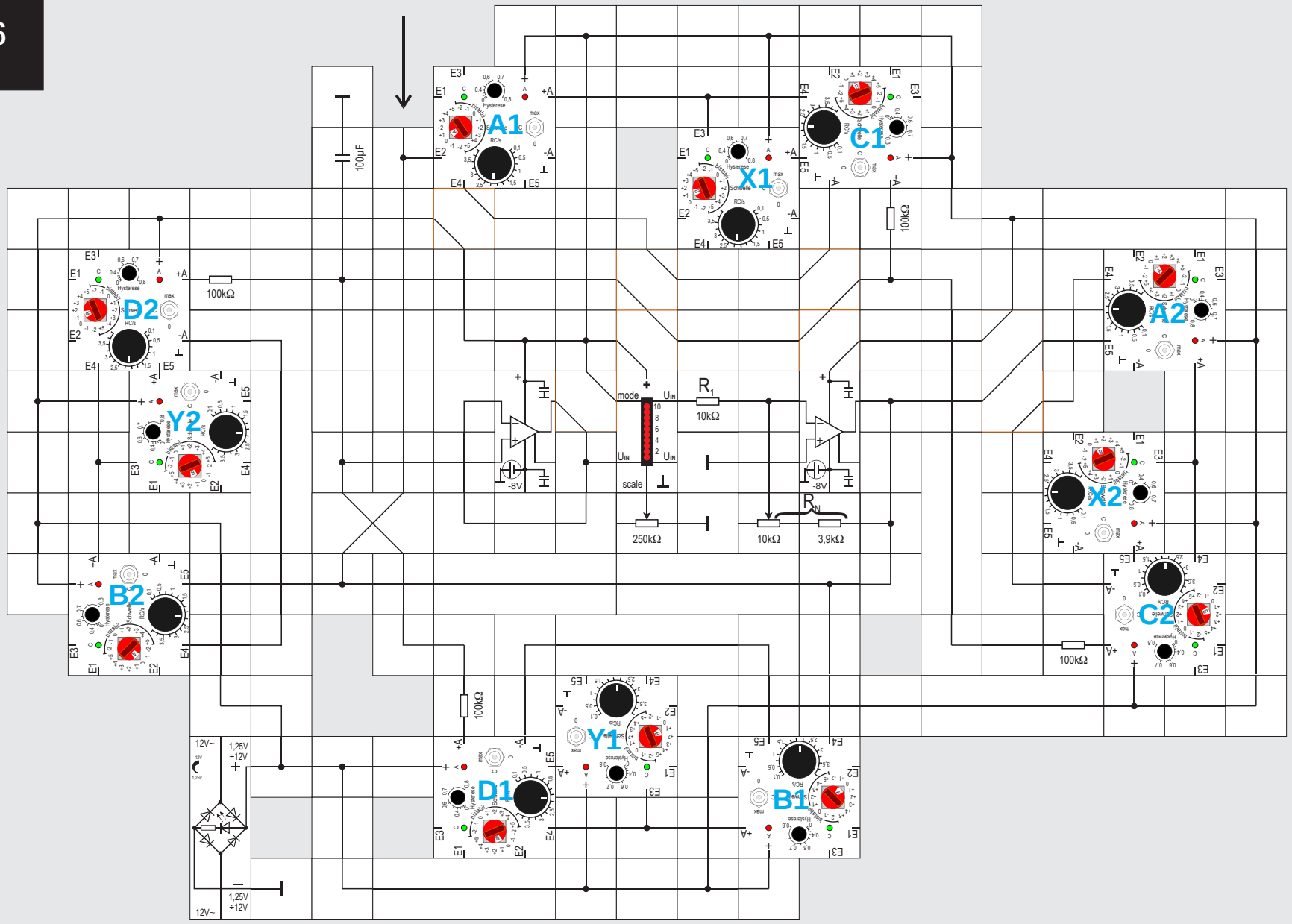
The previous experiment shows nicely how synchronization occurs without a central clock interfering. The correcting signal is gained from the four equal oscillators. The central module K is only used because it contains the required summing unit and comparator. Depending on the oscillator's activities, the summing unit calculates the input signals 0, +1, +2, +3 or +4, which are compared with our threshold by the comparator and then result in further activities or not. So there are five possible concentration levels in the fluid, and the correcting signal is a binary »present« or »not present«, and the length of the signal depends on the randomly active congruence of these four signals.

Those who are annoyed that nature doesn't use levels can refine the model by »rounding« the output signals of the oscillators on their way to the central threshold module K using an RC element (10kΩ/10µF) for each. These modules will be implemented in the setup instead of a line connection. Four diodes must also be added. They ensure that the capacitor charges via the 10kΩ resistor but discharges only via the input resistance of the threshold module with 100kΩ each. The discharge, therefore, will be a lot slower, and the chance that all of the signals are active for a while is bigger.

In addition to that, we can connect the fifth free input of the threshold module to a voltage that is continuously adjustable with the potentiometer, so the threshold is, due to the bias, continuously adjustable, too. We can expect the synchronization to occur faster now.

Anyway, the threshold module still sends a binary synchronization signal, which is not quite a change in concentration in the biological example.

The next setup shows how we can improve this, too. We don't need the RC modules for that, but rather an operational amplifier.


A1
X1
C1
D2
A2
Y2
X2
B2
C2
Y1
D1
B1
100µF
100kΩ
100kΩ
100kΩ
100kΩ
mode
scale
R1
10kΩ
250kΩ
10kΩ
3,9kΩ
RIN
-8V
-8V
12V~
12V~
1,25V
÷12V
1,25V
÷12V

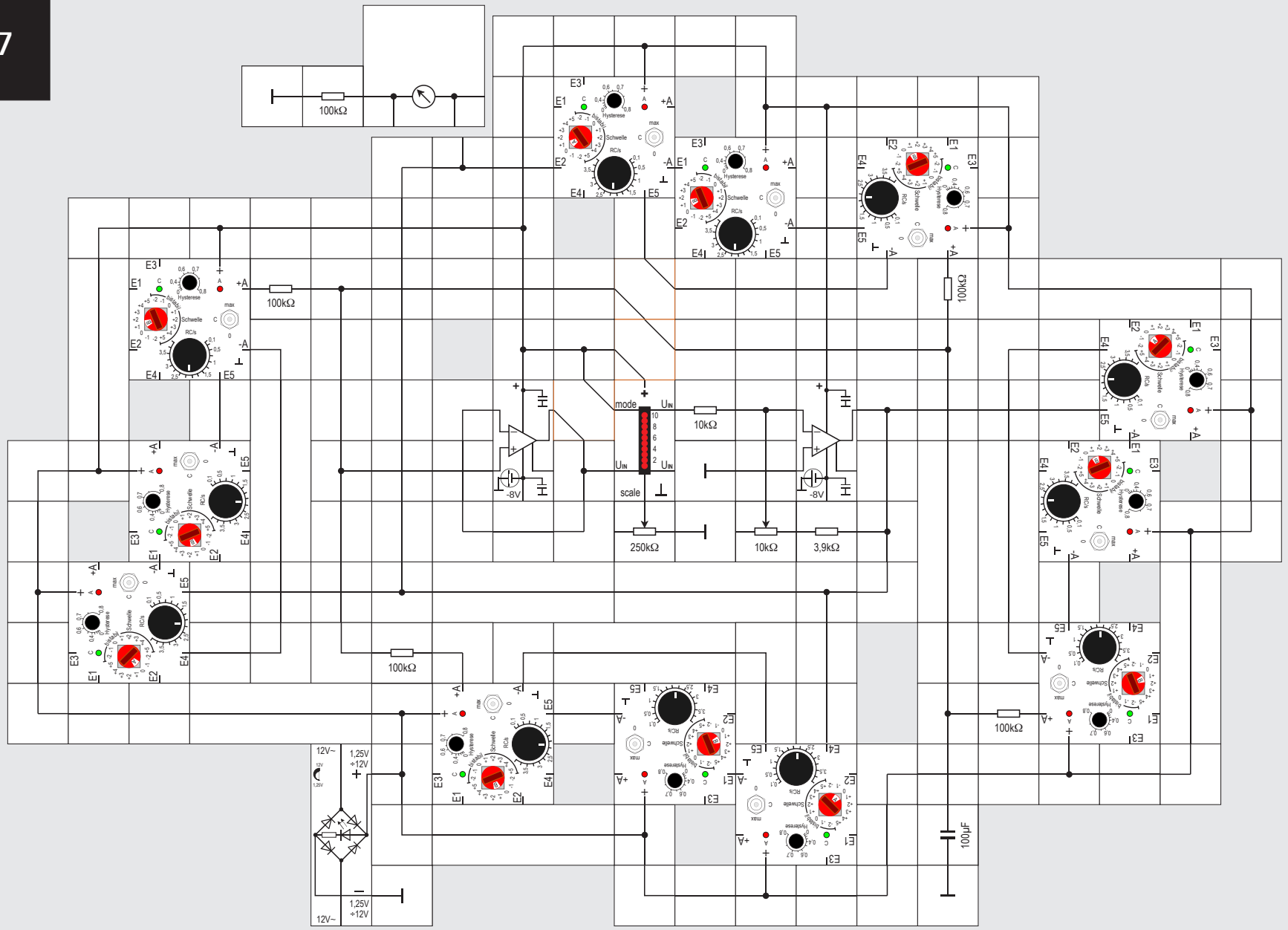
Experiment 46
Improved model II for quorum sensing

We get another improvement of the model by sparing the threshold module K completely and instead letting the concentration increase continuously. Thus, all four oscillators' output signals are connected via one 100kΩ resistor each to a common 100µF capacitor, which charges and discharges slowly, depending on the output signals, as if it follows an exponential function. The (continuously increasing or decreasing) voltage of the capacitor therefore is a measure of the concentration in the common fluid.
This voltage will be amplified non-invertingly by an operational amplifier used as an electrometer (the gain, e.g., amplification factor, is only 1, though) and shown by a light-strip indicator. Per active oscillator signal, two LEDs of the indicator light up, and at maximum, eight of them turn on, if first the 250kΩ potentiometer module, including the ground module, is removed. The operational amplifier is required as an impedance converter. Here, a short explanation should suffice: We want the capacitor only to charge and discharge via the 100kΩ resistors and its voltage should not be influenced by the quite low-impedance resistors of the second connected operational amplifier. This is what we will get by using the operational amplifier as an electrometer. The function and properties of operational amplifiers and the display module are explained in boxes on the following pages for those who are interested.
The second operational amplifier amplifies invertingly with gain factor v = -RN/R1 so a negative signal results.
We can adjust the gain factor at the 10kΩ potentiometer such that $-1.39 \leq v \leq -0.39$. Its output signal represents the capacitor's voltage and therefore is continuous without jumps, too. It is connected to the four oscillators, each of them comparing it with its threshold.
After applying the supply voltage, we turn on the modules C1, C2, D1 and D2 to the »max« position. All oscillators now send an output signal and the capacitor charges to its maximum. Eight LEDs light. Those who want to use the complete light strip can use the potentiometer and adjust it.
At the 10kΩ potentiometer, we can, as already mentioned, adjust the gain of the second operational amplifier. If the rotary knob is at its clockwise maximum, RN together with the fixed resistance equals 13.9kΩ, so our gain is

$$v = -13.9k\Omega/10k\Omega = -1.39$$

At the other end, we only have the fixed resistance, so we get $v = -3.9k\Omega/10k\Omega = -0.39$

If we release the oscillators after the last adjustment by switching them one after another from »max« to »C«, the oscillations are in no way synchronous and will not become synchronous. The correcting signal for its amplitude is too weak to have an effect. The arrow in the setup marks the place where we can measure the (negative) correction voltage with an external instrument like the LECTRON measuring instrument (not included in this experimental kit) in series with a 100kΩ resistor.
It is very different from the previous one if we turn the rotary switch to the clockwise end. Now the signal has a sufficient amplitude, and the synchronization occurs a lot faster than in our first setup. This is because there is no threshold to be reached in the central module before a correction signal is sent; instead the signal is always more or less present and therefore can take effect earlier. Intermediate adjustments allow observing at what point synchronization doesn't work anymore.
If we remove the ground connection of the capacitor for a moment, it establishes similar conditions to those in the first setup; the capacitor is disabled. The light strip shows only five different possibilities; this time, there is no summation, but by voltage separation, only 0%, 25%, 50%, 75% or 100% of the amplitude of an output signal is reached.


100kΩ
mode
scale
UIN
10kΩ
250kΩ
3,9kΩ
-8V
12V~
1,25V
÷12V
100µF

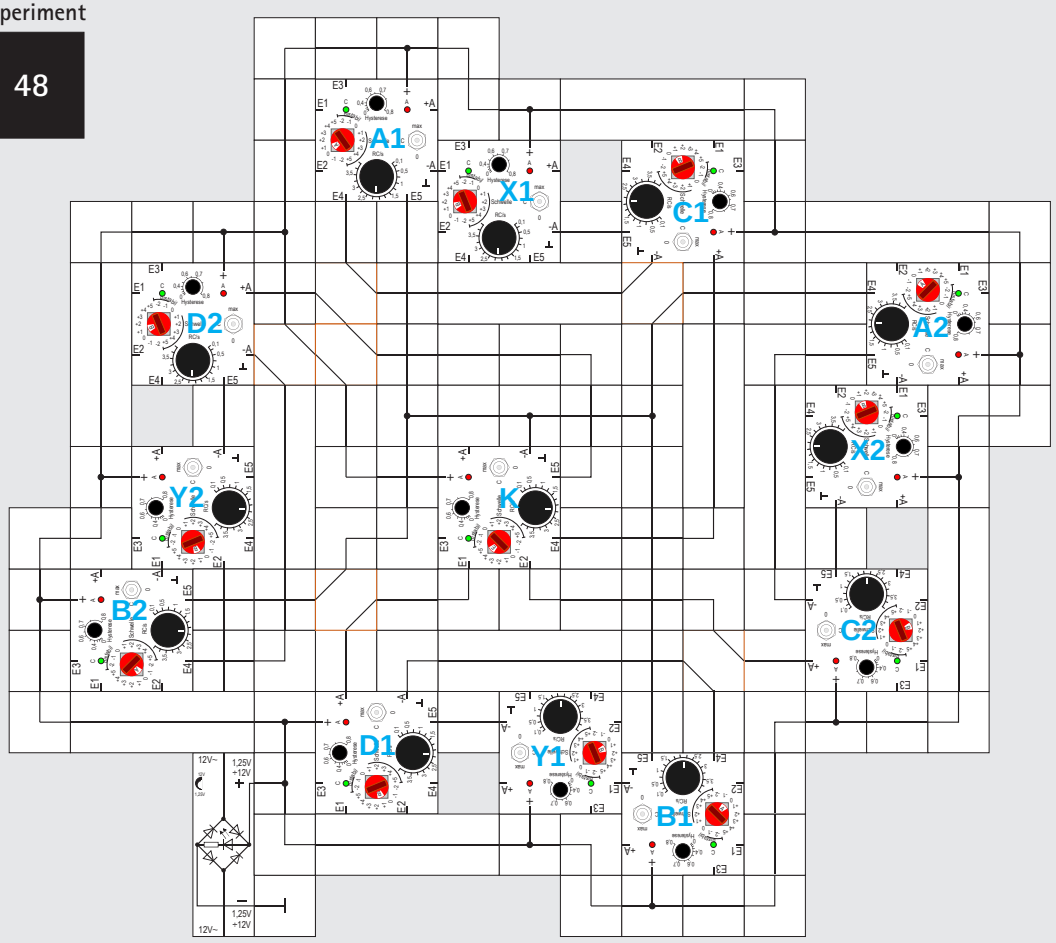
Experiments 47 & 48
Repressilator and quorum sensing

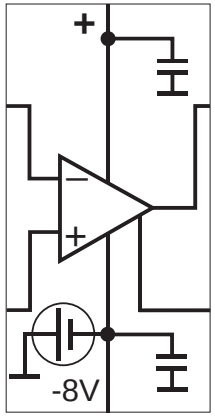
Experiments on quorum sensing are not limited to oscillators that work with the feed-forward principle. In Garcia-Ojalvo et al. and Hellen et al. [20,21] it is explained how multiple repressilators, which we know from experiment 16, can be coupled. The upper figure shows the improved circuit with operational amplifiers and LED display. The shared 100µF capacitor whose voltage symbolizes the concentration of the autoinducer is moved to the lower right. If needed, it is possible to display again the (negative) amplitude of the synchronizing signal with an external voltmeter or the LECTRON measuring instrument (not part of the construction kit). For reasons of completeness, a similar circuit with an extra threshold module K (small picture on the next page) is given. The synchronization properties of this experiment are the same as in the other experiments with oscillators.

If you are interested in electronic engineering and want to learn more about the new modules (operational amplifiers and LED display), you can read more about them starting in the box on the next page. But it is not essential to know all the details, either. Operational amplifiers were also used in [21]. The following experiments show how versatile the LECTRON threshold module is and demonstrate that typical digital circuits like a counter or a shift register can be built.

Experiment

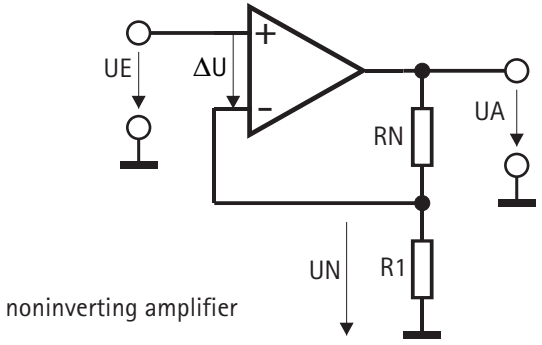
48





Operational amplifiers in a nutshell

Operational amplifiers (also called op-amps) are the most commonly used modules in electronics.
The LECTRON op-amp module contains an operational amplifier, which is the integrated circuit CA 3140. Both left connections are its inputs; the one above is the inverted (-), and the one below is the noninverted (+). The output of the amplifier (top of the triangle) is at the upper right connection.
Operational amplifiers usually need two current supplies, a positive and a negative one, so they can handle negative input voltages, too. The LECTRON op-amp module only needs a positive one, which is connected to the narrow upper side; the negative one is produced internally by a charge pump (described two pages ahead). The voltage (-8V) on the narrow side on the bottom can be used for certain applications; in our experiment, it



noninverting amplifier

is not needed, nor is the bottom right contact. Both remain unused.

The operational amplifier is a differential amplifier that amplifies the difference between the potentials U = (U_+ - U_-) of both input signals; they are labeled with + and -. This amplified voltage is available at the output (in the circuit, the output at the top of the triangle). Therefore it is not important if U_+ = 5.1V and U_- =5.0V or if U_+ = 4.6V and U_- = 4.5V; the operational amplifier always amplifies the difference ΔU = 0.1V.

The input labeled with »+«, is the »noninverting« input, and the one labeled with »—« is the »inverting» input. Noninverting means that with a constant voltage U_- a rising voltage at this input leads to a rising voltage at the output. Inverting means that the voltage at the output decreases if with a constant U+ voltage the voltage at this input is increasing. All in all, the operational amplifier executes the mathematic operation

$$U_{output}=[U_{+input}-U_{-input}] \times \text{amplification-factor}$$

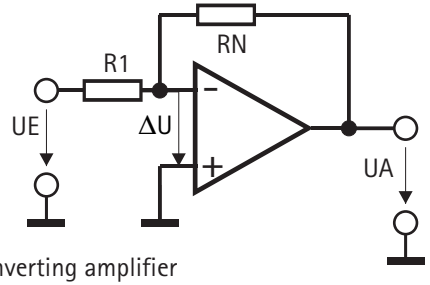
It subtracts the voltage at the inverting input from the voltage at the noninverting input and multiplies the result by the amplification factor (which is called the gain).

Modern operational amplifiers have extremely high input resistances in the TΩ range. $1T\Omega = 10^{12}\Omega$; i.e., there is literally no input current into the module. Also, the so-called »open loop gain» factor of the operational amplifier (no feedback from its output to its input) is, with typical values from 10^4 to 10^5, enormously high. That doesn't mean that with ΔU = 1V the amplifier produces an output voltage of $U_A = 10^4$ V; U_A is of course only able to swing between the two supply voltages. The positive voltage of the current supply cannot even be reached because of the circuit design. U_A of the LECTRON module, which gets the supply voltage of 9V, is for example $-8V \leq U_A \leq 7.5V$.

An operational amplifier with no feedback is qualified to be a so-called comparator, which compares U_+ and U_- with each other.

If $U_+ > U_-$, the output goes in positive saturation, and for example produces +7,5V; if $U_+ < U_-$ then U_A = -8V.

The operational amplifier would not be such a commonly used module if it were used only as a comparator. In most applications it has (negative) feedback. In the previous experiment, we already used the extremely high resistance feature of its inputs. It worked as a so-called electrometer-amplifier in the noninverting mode.



inverting amplifier

To grasp the behavior of an operational amplifier with external negative feedback, two »golden rules« suffice for almost everything you'll likely ever encounter. First, as we already mentioned, its voltage gain is so high that a fraction of a millivolt between the input terminals will swing the output over its full range, so we ignore that small voltage and state golden rule number 1:

The output attempts to do whatever is necessary to make the voltage difference between the inputs zero (ΔU =0).

Of course, this doesn't mean that it changes the voltage at its inputs. It can't do that. What it does is »look« at its input terminals and swing its output terminal around so that the external feedback network brings the input differential to zero (if possible).

Golden rule number 2: Operational amplifiers draw no input current.

These two rules get us quite far. So let's begin with the inverting amplifier circuit (see the right figure on the preceding page). The analysis of such a »tamed« operational amplifier is simple:
The noninverting input (+) is at ground, so rule number 1 implies that the inverting (−) is also because $\Delta U = 0$. It is called a »virtual ground« at the (−) input. This means that the voltage across RN is U_A and the voltage across R1 is U_E. So using the second rule and Kirchhoff's current law (the sum of all currents equals 0) we have

$$U_E/R1 + U_A/RN = 0 \text{ or}$$
$$U_A = -U_E \cdot RN/R1$$

In other words, the voltage gain of the inverting operational amplifier equals

$$v = U_A/U_E = - RN/R1$$

The input resistance or impedance Z_{in} is easily calculated. The inverting input (−), the virtual ground, is always at 0V. So $Z_{in} = R1$. We cannot figure out the output impedance for this circuit, but it's a fraction of an ohm.
And now to the left figure on the preceding side. It shows the circuit of the noninverting amplifier. Again, the analysis is simplicity itself:

$$U_N = U_E \text{ because } \Delta U = 0.$$

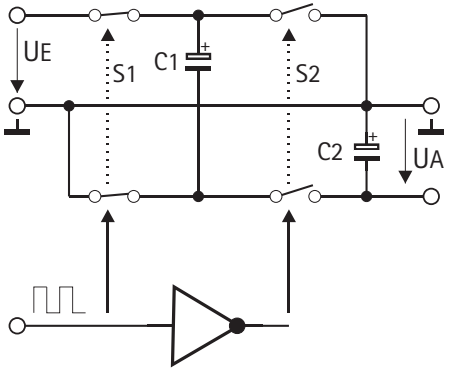
But U_N comes from a voltage divider consisting of RN and R1. It works in no-load condition because there is no input-current at an ideal operational amplifier. So:

$$U_N = U_A \cdot R1/(R1 + RN)$$
$$U_A = U_N(1 + RN / R1)$$
$$U_A = U_E(1 + RN / R1)$$

The voltage gain $v = U_A / U_E$ is revealed as:

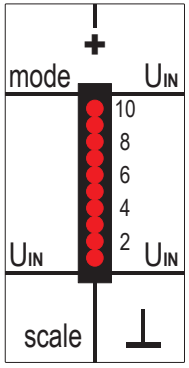
$$v = 1 + RN / R1$$

This is the noninverting amplifier. In the approximation we are using (no input current), the input impedance is infinite. The output impedance is still a fraction of an ohm. The circuit is also called an »electrometer amplifier« because it is suitable to measure potentials of static electricity experiments with its high impedance sources.
With RN = 0 and R1 = ∞ there is only a voltage gain of $v = 1$ and thus just a change in the impedance. This circuit is called then a »voltage follower«.
Both basic circuits are expandable to more inputs, and the result is a summing amplifier like used inside the circuit of the gene module or a differential amplifier.



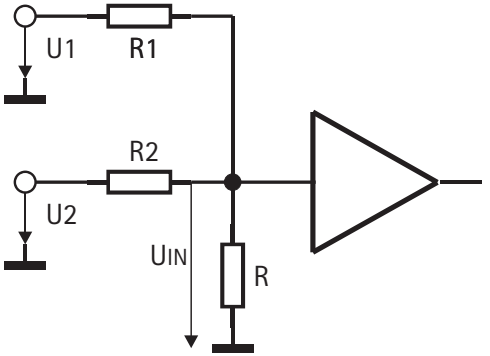
The charge pump

Inside the LECTRON op-amp module, the negative supply voltage is generated with a so-called charge pump, an integrated circuit that is a switched-capacitor voltage converter. Therefore, a capacitor C1 (see figure) is connected to the voltage of the positive current supply, charged, and then connected by the (electronic) switches S1 and S2, which are inversely phased by a free-running oscillator to the capacitor C2 at the output with reversed polarity. After a few cycles, UA = -UE. C2 filters the output ripples. There is a small voltage drop proportional to the load current at the electronic switches and a loss at the capacitors due to the charge reversals, so UA barely reaches UE, but is typically less than 1V. The loss can be kept small by using high capacitor values and switching frequencies.



$U_{full\ range}$/V	R_{scale}/Ω
7,5	∞
6	33k
5	15k
4	7,5k
3	3,9k
2	1,4k
1,5	430
1,25	0

R_{scale} between scale and ground



The display module

The LECTRON LED-display module was developed as a low-cost alternative to the oscilloscope. Certainly it cannot replace the oscilloscope, but it displays fast signal changes better than a measuring instrument can. The module contains a bar display consisting of 10 LEDs. The higher the voltage at U_{IN} is, the more LEDs are turned on. The measuring-range (the »full-scale deflection«, at which all 10 LED's are turned on) is adjusted with the resistor from »scale« to ground. The table shows the resistors used for some common measuring ranges. With the scale-input open ($R_{scale} = \infty$) the tenth LED is flashing if $U_{IN} \geq 7.5V$. If we choose for example $R_{scale} = 3{,}9k\Omega$, this happens already at 3V. For those who want to calculate on their own:

$$U_{full\ range} = 1{,}25V \cdot [1 + 5 \cdot R_{scale}/(10k\Omega + R_{scale})]$$

If the exact value is not of interest, it is recommended to use a 10kΩ/250kΩ potentiometer or a 10kΩ adjustable resistor. We then can adjust the measuring range continuously. The single pin »mode« changes the display from a moving dot to a bar graph. If it remains open, just one LED is turned on as a dot; connected to the power supply, we get a bar display.

The input current (maximal 100nA) doesn't much affect the potential of the circuit. Therefore we can directly display voltages from high-impedance sources. If a display of only the AC parts of a signal is needed, it is recommended to use an RC combination (47nF/100kΩ).

The module, however, can only display positive voltages. Shifting the input signal in question to positive ranges is simply done by adding a constant voltage. Due to the high impedance inputs, we can also use high-impedance resistors to build up the voltage divider. The three linked U_{IN}-inputs of the module help save connection modules.

The sum of all currents in the crosspoint is zero (Kirchhoff's law):

$$(U1-U_{IN})/R1 + (U2-U_{IN})/R2 = U_{IN}/R$$

transposing to U_{IN} we get the somewhat complicated equation

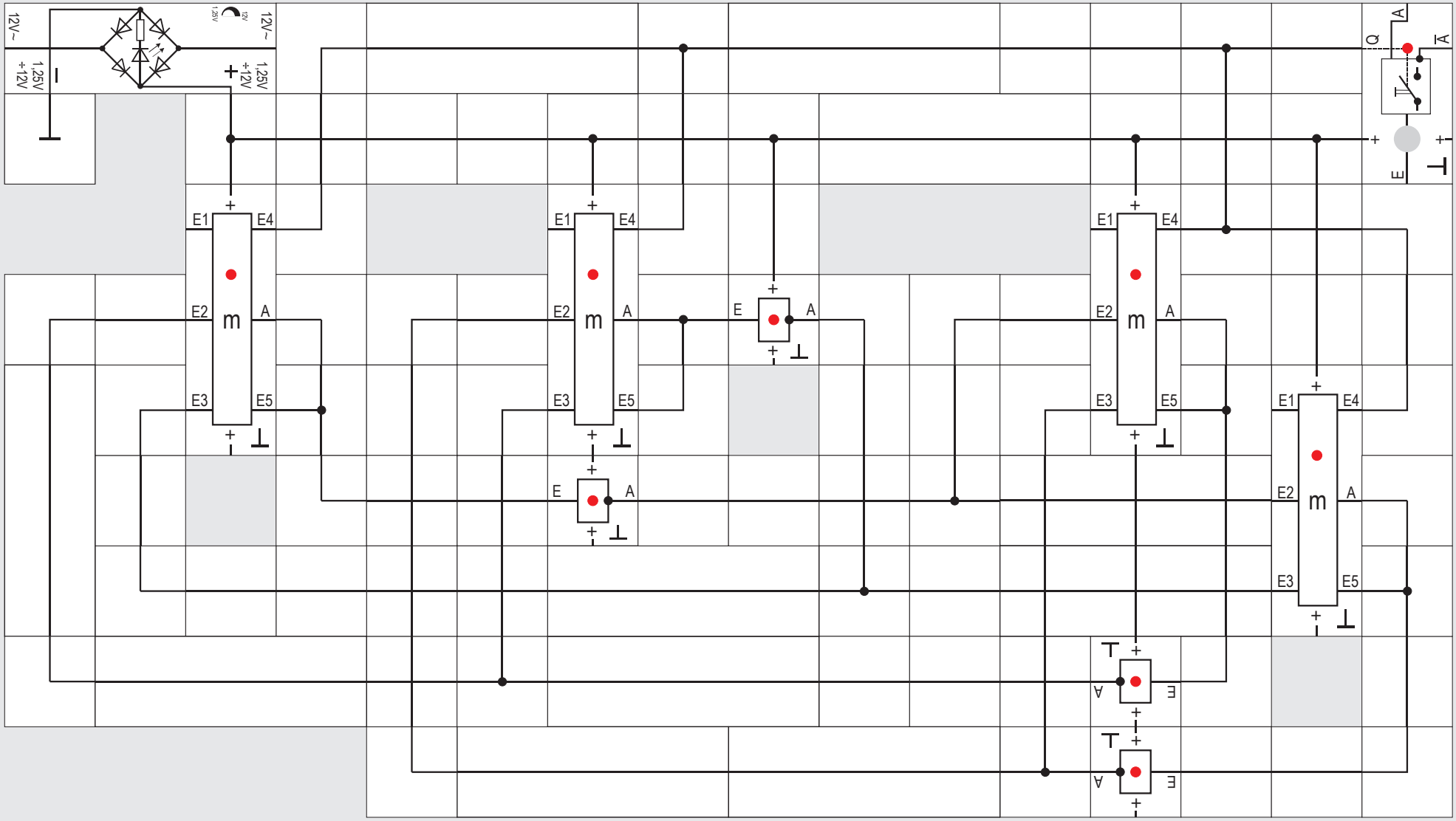
$$U_{IN} = (U1 \cdot R/R1 + U2 \cdot R/R2)/(1 + R/R1 + R/R2).$$

That becomes simpler with R1 = R2 :

$$U_{IN} = (U1 + U2)/(2 + R1/R)$$

and if we omit R completely : $R = \infty$

$$U_{IN} = (U1 + U2)/2$$

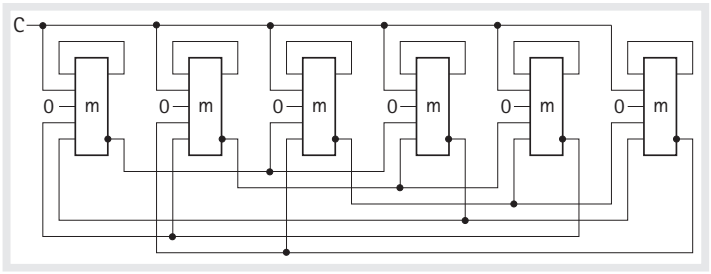

12V~
1,25V
÷12V
E1
E2
E3
E4
E5
m
A
E
Q
Ā
T
E
⊥

6. Digital circuits with the gene module

You will surely have noticed that the setups we built with gene modules now sometimes look a little like computer circuits. For those who are interested, we do not want to withhold further gadgets. We can assemble more real basic circuits from digital technology with the gene module. For all we know, the following setups have never been found in biological gene networks, but isn't it fascinating that gene modules are able to count?

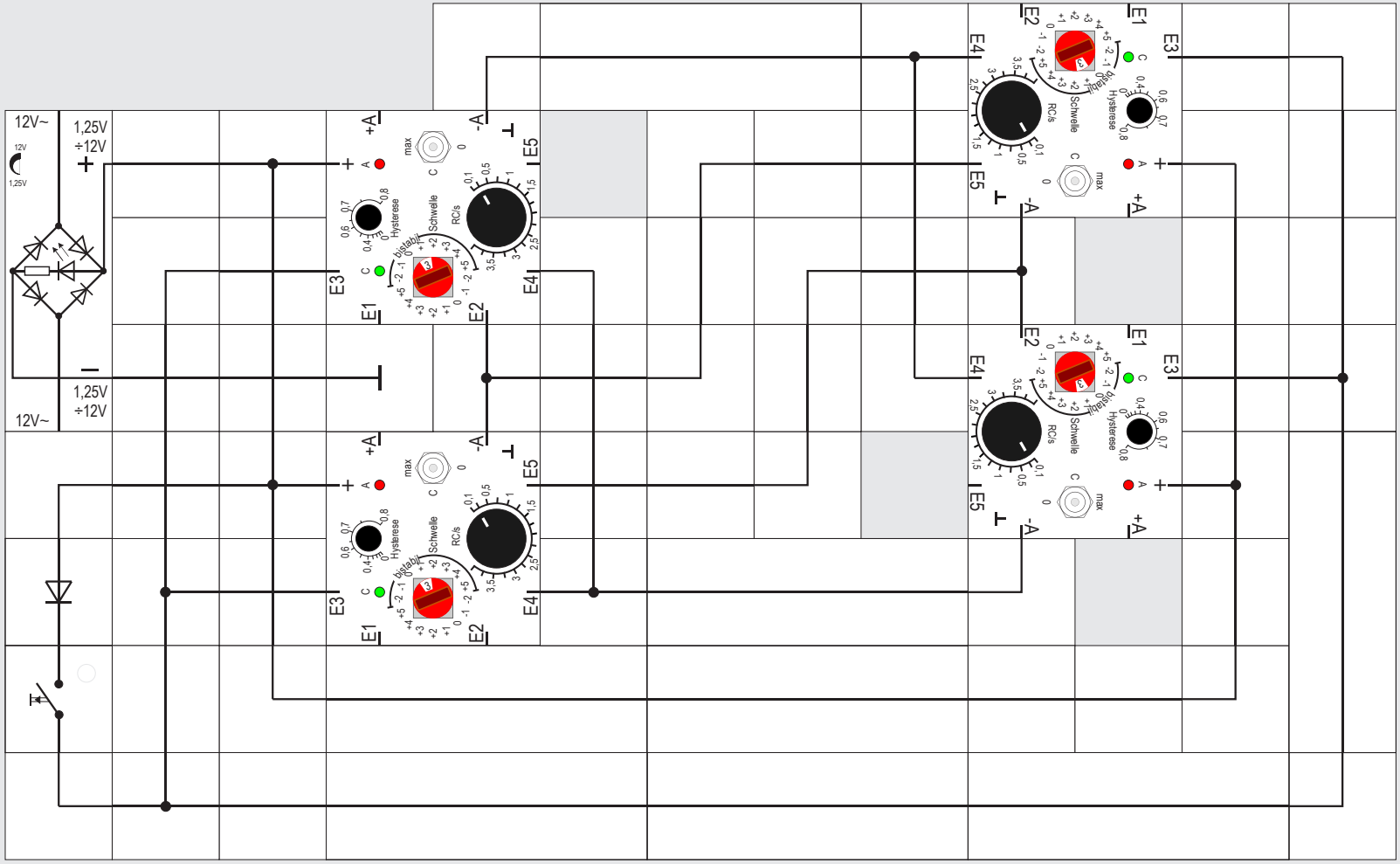
Experiment 49
Counter with 5-input-majority modules

There is a digital technological sister of the gene module in the repertoire of the LECTRON modules. That is the majority module with 5 inputs, which we want to introduce for comparison: The five inputs receive a logical 1 or 0 signal, and the signal forming the majority will be sent to the output of the module. In the guidebook of the experimental kit »Threshold and Majority Logic«, we find a counter built with this module, and it is interesting to find out if this is possible with the gene module, too. Before we try, here is the description of the experiment from the sister kit. The counter has the structure of a ring, as we know from previous experiments. Its counting volume is the number of its cells, N; the complexity of a ring counter is higher than that of a usual counter. The pattern of the setup only applies to counters with an even number N and $N \geq 4$ [22].



One input of each module is fixed to 0. The second input is connected to the central clock signal C, and the output A of each module is fed back to the third input of the same module. Both still available free inputs of each module are connected to the inverting outputs $\overline{Ak}$ of previous cells. The following connection rule applies:
The inverting output $\overline{Ak}$ of kth cell with $1 \leq k \leq N$ is connected to an input of the cell k+n and another one of the cell k+n+1whereby n = N/2 - 1. If there is no following cell, just start from the beginning. Finally, we have completed a ring structure. Maybe the rule sounds like a mystery, but when you look at the figure for a counter with six cells we are quite sure you will grasp the blueprint.

The counter counts to 6, so N =6 and n = (6/2)-1 = 2. The inverting output of (k=1)th $\overline{A1}$, the first cell, is connected to one input of the third and to one input of the fourth cell, and so on (see figure).
With four 5-input-majority modules, we can count up to four. The inverting output of the first module is connected to an input of the next module and to an input of the module after the next, because according to the formula, n = 1, and so on. If C = 0, n adjacent modules send a 1; if C = 1, another 1 is added, and when C = 0 again, one of the modules that sent a 1 before now becomes 0, so n 1's went a step/module forward. We don't have to build this counter; in particular, we are missing modules to do so (4x inverter no. 2450, 4x majority module no. 2452). It is sufficient to understand its function because we want to translate it into a gene module circuit.


12V~
1,25V
÷12V
12V
1,25V
+
–
1,25V
÷12V
12V~
+A
-A
E1
E2
E3
E4
E5
max
Hysterese
Schwelle
RC/s
bistabil
kippfähig

Experiment 50
Counter with four gene modules

Our gene module with five or six inputs includes the majority function with five inputs (see experiment 6). So it must be possible to transform the setup of experiment 49 to one with gene modules. Thus we look at a single module and the connections to its neighbors and »translate« it to threshold logic.

- The threshold of a majority module with five inputs is 5/2 = 2.5. Three logically identical variables determine the output. The gene module has the same function if its threshold is set to +3.
- One input of the majority module is fixed to 0. So we simply leave one input of the gene module open.
- The second input is connected to the module's own output +A. We adjust the threshold to »+3 bistable« and save the external connection (see experiment 4).
- Now it gets a bit more difficult with the other input signals coming from $\overline{A}$-outputs of two other modules. We don't have $\overline{A}$-outputs in our threshold modules, we have got only -A output signals, which we already know is a big difference. The table clarifies it again.

	logical 0	logical 1
+A	0V	+8V
$\overline{A}$	+8V	0V
-A	0V	-8V

If we don't have an inverter or don't want to use it, the following basic relation will help us again; we remember that it is always true in Boolean and threshold logic, too:

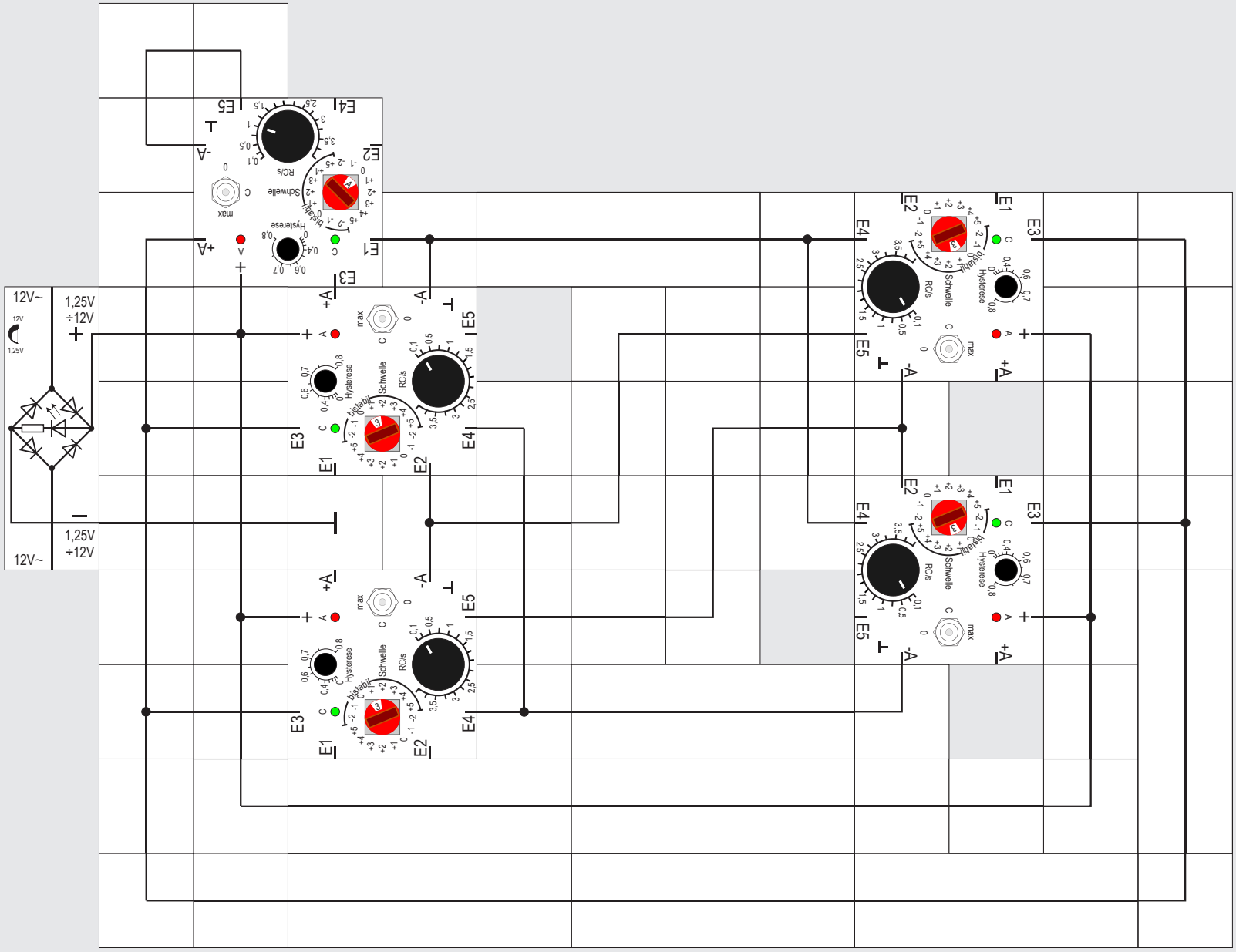
$$A + \overline{A} = 1 \text{ or } \overline{A} = 1 - A$$

We can create a $\overline{A}$- signal by fixing one of the gene module's inputs to logic 1 (+8 V) and a second one to -A which fortunately exists. We would have to generate both constant signals of logical 1 with a combination of diodes and resistors out of the supply voltage. But this is not necessary, because we can compensate for both constants by simply omitting them and decreasing the thresholds by 2.

We generate the central clock C from the supply voltage (which is decreased with a diode) with a push-button module.

The button doesn't have to be debounced, because we have to press it for a while anyway. We choose the time constant RC = 0.1s and hysteresis 0.1 for all modules.

If we apply the supply voltage, nothing will happen. Only if we push the button two adjacent modules turn on. It is random which ones these are. After releasing the button, one of them (the one in the front, counterclockwise) remains turned on. When the button is pushed again, the next module, counterclockwise, turns on; we have to push the button until this happens. Two modules are now turned on, the previously active one turns off when the button is released. In this way, we can shift the active module one step counterclockwise in the ring.


12V~
1,25V
÷12V
+
–
1,25V
÷12V
12V~
E1
E2
E3
E4
E5
+A
-A
max
Hysterese
Schwelle
RC/s
bistabil

Experiment 51
Counter with four gene modules, alternative setup

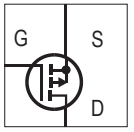
If we use an additional gene module working as an oscillator, we can do without the manual clock function. The circuit works just like in the previous experiment. We replace the button with the well-known oscillator. For reasons of space, the E3 input is connected to the +A output of a cell, which of course would stop the oscillation. We compensate for this by connecting the -A output with another input (E1).
We remember another rule that can be useful sometimes, too: Unneeded inputs can remain unconnected or be connected to unused inputs of different gene modules. No unwanted couplings will occur.

E1
E2
E3
E4
E5
+A
-A
max
Hysterese
Schwelle
RC/s
bistabil
12V~
1,25V
+12V
-12V

Experiment 52
Counter with six gene modules

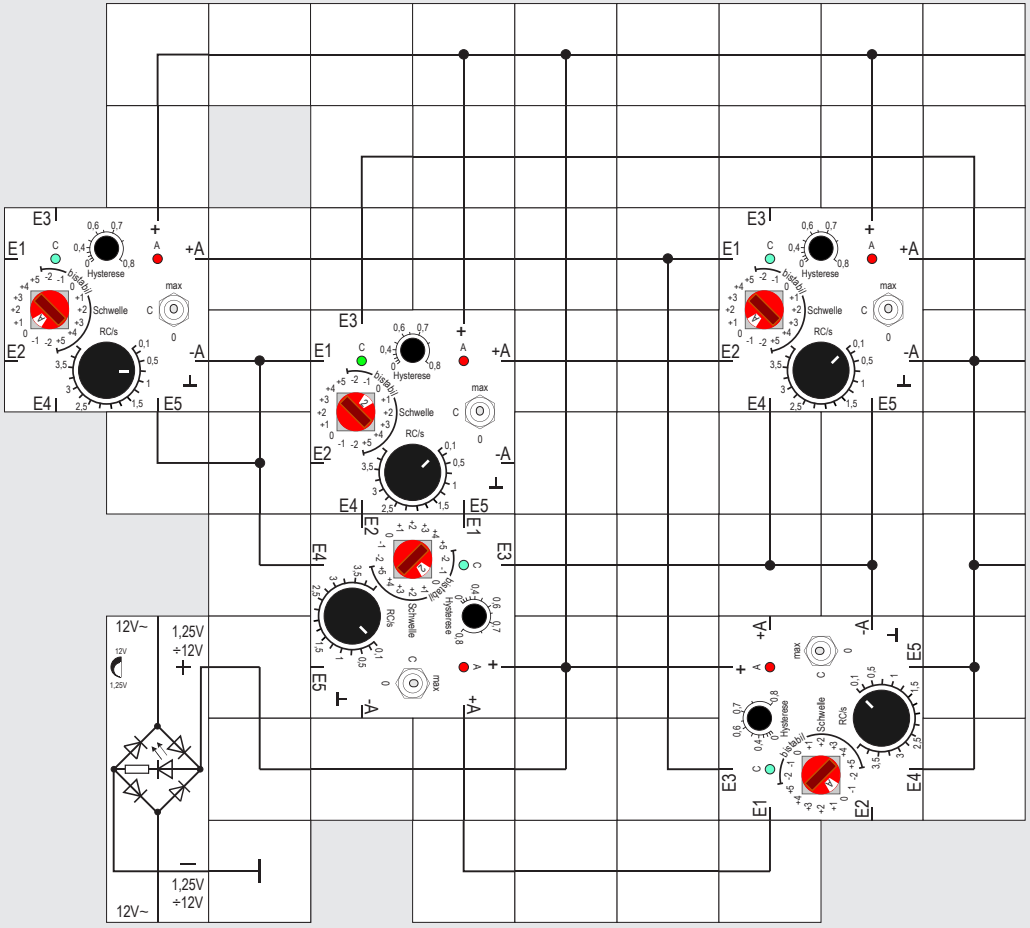
Finally, on the matter of counters, we show the setup with six gene modules. This is a version in which the clock is generated manually with the press-button module.

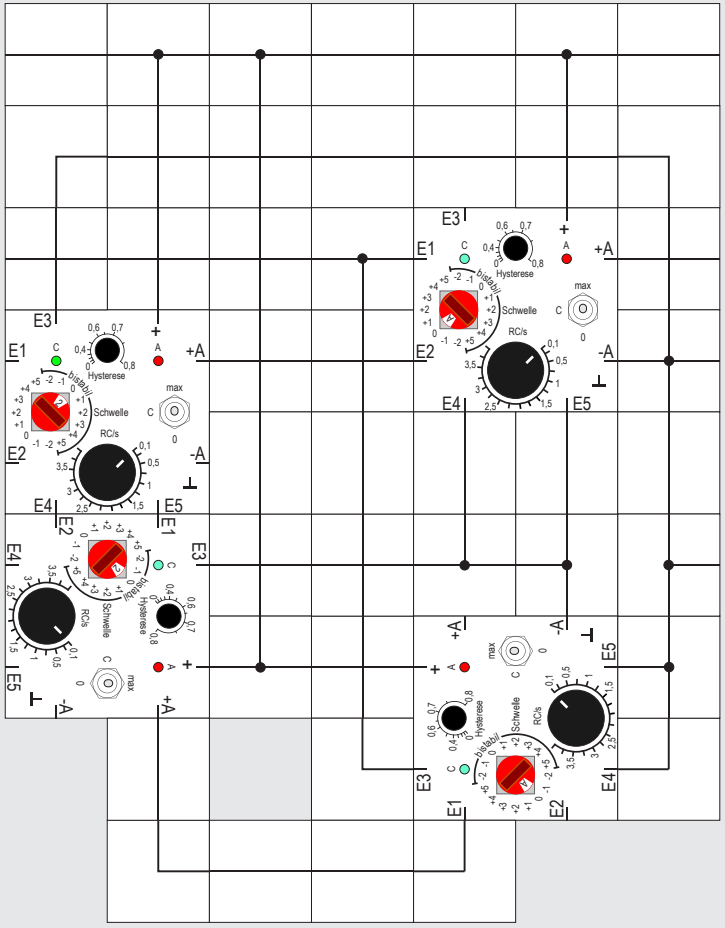
For those who are interested in electronics



The MOSFET

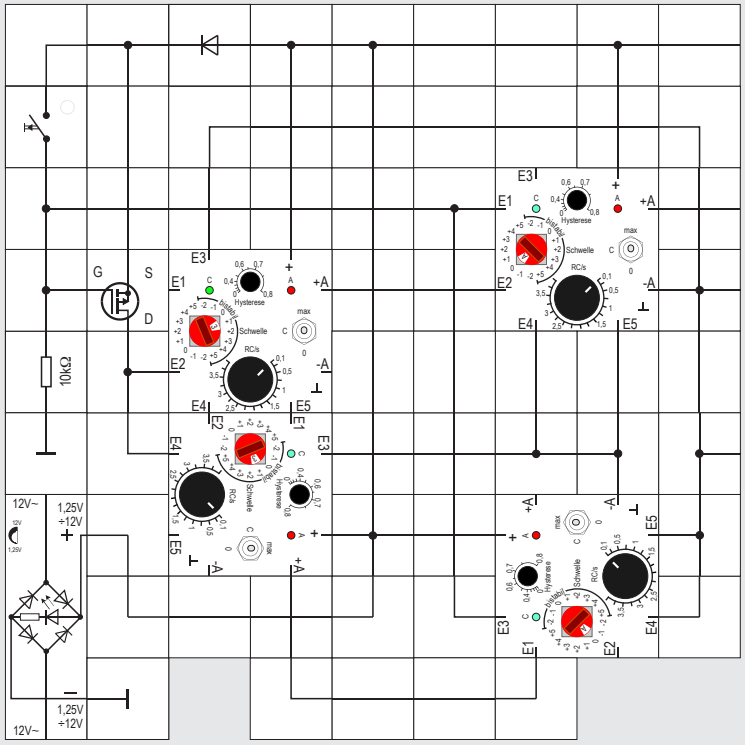
In the experiment after the next, we will generate a clock manually with the push-button module. We have to consider how to do it because we simultaneously need an inverted clock, too.
To get an inverted signal from a given one is pretty easy with a so-called MOSFET (metal oxide semiconductor field-effect transistor). This is a transistor with three pins or electrodes called source S, gate G and drain D. The gate is isolated from both other electrodes and therefore has very high resistance in the TΩ range. We will use a so-called p-channel MOSFET in the experiments. This device forms a well-conducting channel between source and drain if the gate potential is at least ~2V below the source potential. It blocks instead if the gate potential is equal to the source potential.
Therefore the transistor behaves like a switch between the electrode source and drain, which can be controlled, e.g., opened and closed, only by the gate potential. Due to the extremely high impedance, there is no gate current if it is turned on or off.

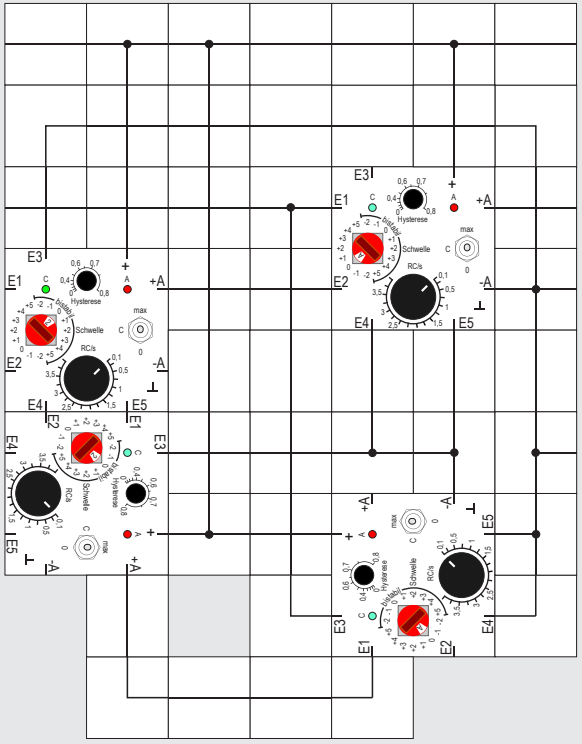

E1
E2
E3
E4
E5
+A
-A
Hysterese
Schwelle
RC/s
max
12V~
1,25V
÷12V

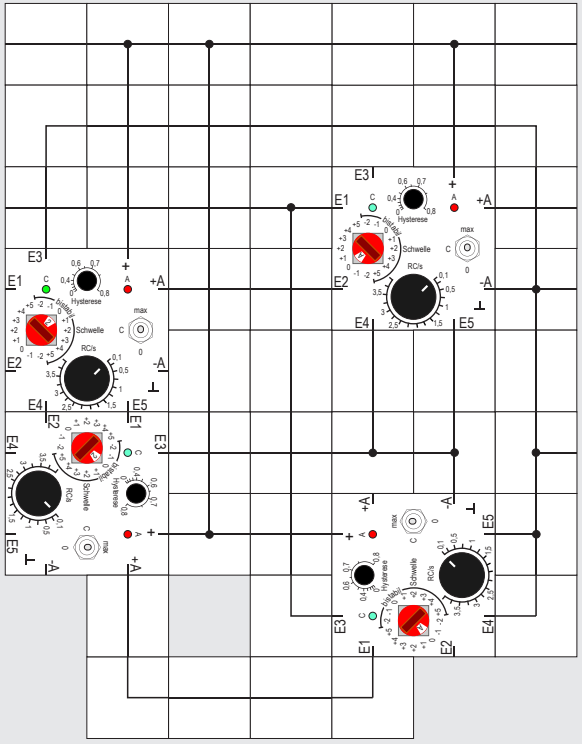

E1
E2
E3
E4
E5
+A
-A
Hysterese
Schwelle
RC/s
max

Experiment 52A
Counter stage

It isn't very common to choose a ring structure for counting rather than another one that is less complex. If we succeed in designing a circuit that divides the clock-controlled signal by 2 and provides this divided signal to its outputs, we already have the basic element of a binary counter.
From threshold logic [5], we know digital circuits of different types of flip-flops. For an asynchronous counter, we can use the T flip-flop (T stands for toggle) and »translate« its circuit to a feasible one with our gene modules. The T flip-flop operates as a clock controlled by the principle of master and slave: The logical state of the slave's -A output, which is also the inverting output of the divider stage, is stored with the negative edge of the clock by the master of the following stage, and then shifted to its slave by the positive edge of the clock. Then the slaves send the inverted state to its outputs. This ultimately means a division of the clock signal by 2. Both output signals of the divider stage are used as clock signals for the next divider stage, and so on, and thus a binary counter results.
For a divider stage, we need four gene modules, two for the master part and two for the slave. Using gene modules that can't interpret edges but compare sums of potentials with thresholds, we need two complementary clocks. It should be obvious that a single clock signal cannot first lift the sum of the potentials over the threshold and then drop it under the threshold.
Both clocks, T and -T, will be generated by the gene module on the left that works as an oscillator. Its -A = -T output signal controls the master, and its +A = +T output signal controls the slave. For the storing function of the master, we use the bistable function of the gene module. The threshold is 0.
The -A outputs of the slave are connected to the master's inputs and, for latching, to two inputs of the other master gene module. The »cross-coupling« of the gene modules of the slaves is weighted twice for the sums, which are each compared with the threshold 0.
The oscillator frequency of the left gene module is adjusted with RC between 0.5s and 1s and a hysteresis of 0.1, so we can observe easily what is happening. The RC adjustment of the other four modules is 0.1s and the hysteresis is 0.
After applying the supply voltage, we will see that when the oscillator's red LED is dark, one gene module of the master is activated, while the other is not. When the red LED of the oscillator turns on afterward, this logical state is transferred to the slave, and both of the master gene modules are turned off. When the oscillator LED then turns off again, the other master module turns on, and this logical state is transferred to the slave as well when the oscillator's LED lights again.
If this is too fast for the eye, we can decrease the oscillator's frequency or even clock manually back and forth with the toggle switch between the »max« and »0« positions. Then again, the frequency should not be too high, so the divider stage can follow.
With the output signals Q0 and -Q0 as the clock, we can add another divider/counter stage of four gene modules and already count from 0 to 3, which means dividing the clock frequency by 4.


G
S
D
10kΩ
12V~
1,25V
÷12V
1,25V
÷12V
12V~
E1
E2
E3
E4
E5
+A
-A
Hysterese
Schwelle
RC/s
max


E1
E2
E3
E4
E5
+A
-A
Hysterese
Schwelle
RC/s
max


E1
E2
E3
E4
E5
+A
-A
Hysterese
Schwelle
RC/s
max

Experiment 52B
Extending the counter

The built-up counter is easily expandable. The only limit is the space on the assembly board and the number of gene modules. Now we want to show that you can also control a clock manually with a button. But first, there is a problem to be solved. Until now, we used the clock signal T between 0V and 8V and a clock signal -T between 0V and -8V. To create the second one, we need a supply voltage of -8V, which we don't have. We can easily create the clock signal +T and invert it with the MOSFET transistor. When the button is open, the gate connection G of the p-channel MOSFET is connected to the ground potential via the resistor. As a result, the transistor is highly conductive and the current flows from the supply voltage across the diode (dropping the voltage by 0.7V) to the inputs E2 and E4 of the master gene modules. But we only have T and not -T. If we remember the basic rule

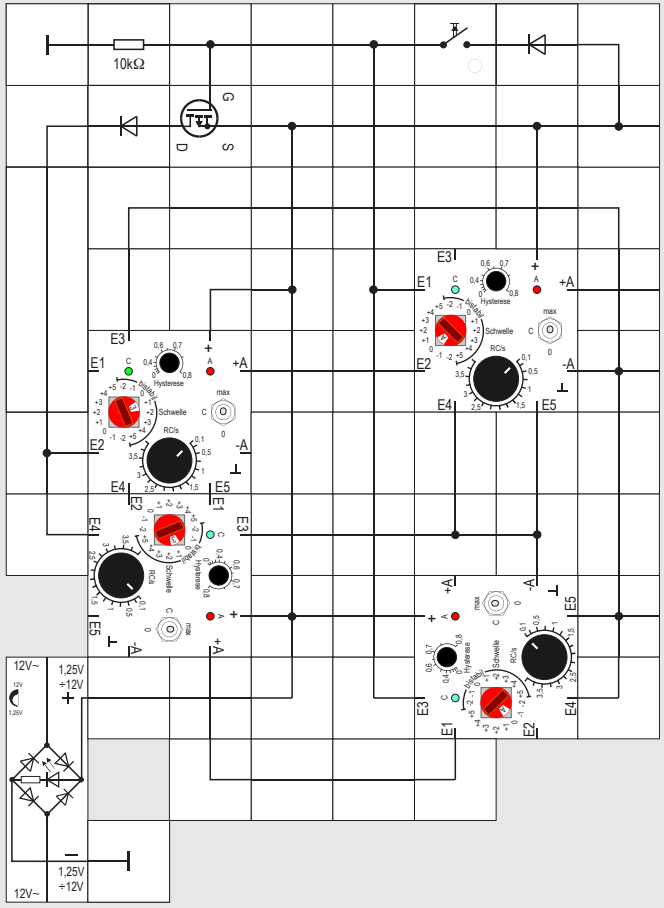
$$T + \overline{T} = 1$$

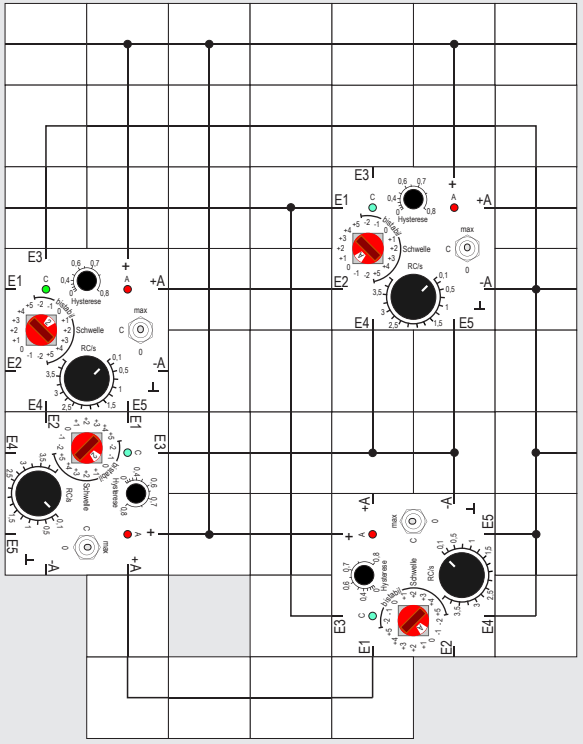
of experiment 50, we get after rearrangement

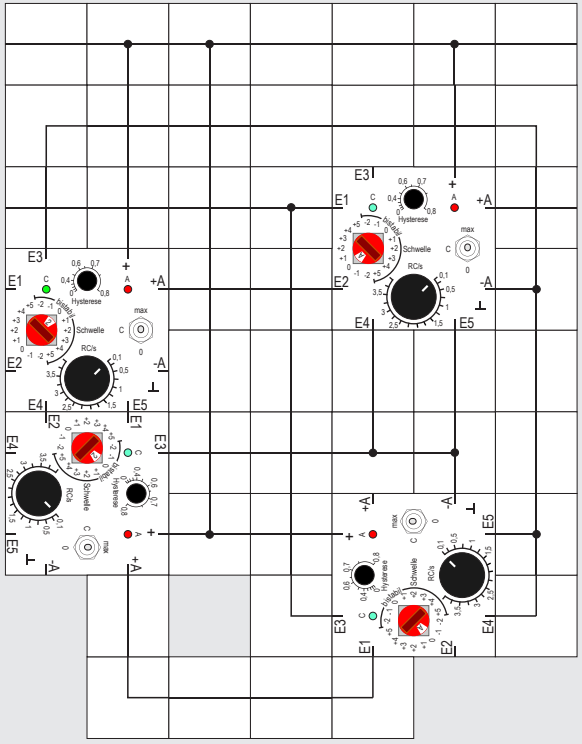
$$-T = \overline{T} - 1$$

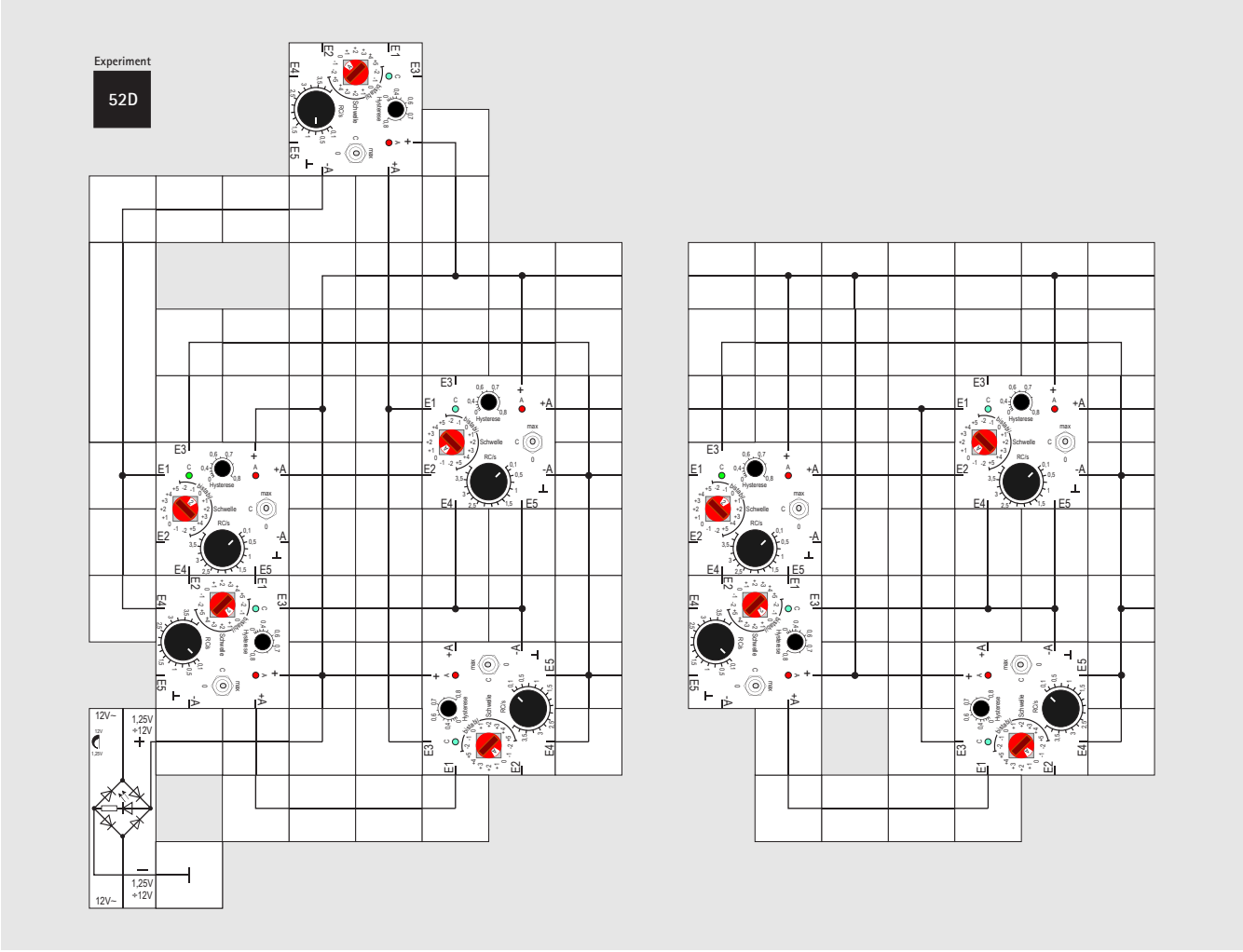
This means, instead of the required -T signal, we can use the $\overline{T}$ signal, but will have to add an additional constant -1(= -8V) signal wherever it is connected to a gene module. It seems like this doesn't help us because a negative voltage is not available without an additional battery module. But we can eliminate a constant minus signal by adding a constant +1 signal and, for compensation, raise the threshold by +1. In the sum, the constant signals remain 0 and don't need any further attention. Only the threshold has to be raised from bistable 0 to bistable +1, and only in the first divider stage receiving the $\overline{T}$ signal. The following stages don't need this because the gene modules can produce the -A (= -Q) signal internally.
The button has to be pushed until the stages switch, so it cannot be released too early. In the setup of the stages, we avoided unneeded inputs of different gene modules being in contact with each other. The internal circuit is immune to these apparent couplings. It can happen that the setup with three stages exceeds the size of the assembly board. Since the right column of connection modules doesn't have any ground connections, it should work, although these modules are not completely on the board. It is important that the ground connection on the far left side is in contact with the assembly board. By pushing the button, we can observe how the setup counts from 0 to 7. The LEDs of the lower slaves are the (binary) counter reading.

52C









Experiments 52C & 52D
Counter stage (alternative set ups)

The setup figure above shows an arrangement with manual clock control that fits on the assembly board without overhang. By using the 13th gene module, we can again generate the clock with an oscillator. The figure on the right shows a possible setup. Don't forget to adjust the thresholds of the two left gene modules back to »0 bistable«. Because the supply voltage module is at its limit, its bottom magnet, which also works as a heat sink, gets quite hot (> 50°C) during long periods of operation. An internal performance monitor protects the module from being destroyed and might shut it off reversibly. The bottom magnet should always be in good contact with the cooling assembly board.

July 29, 1969 K. R. KAPLAN 3,458,734

SHIFT REGISTERS EMPLOYING THRESHOLD GATES

Filed Jan. 31, 1967

Fig. 1.

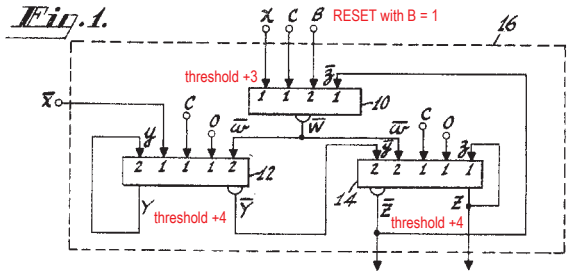


Fig. 2.

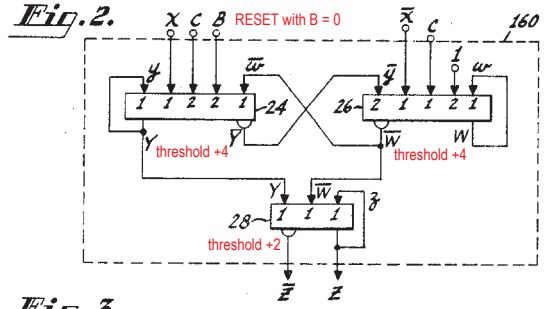
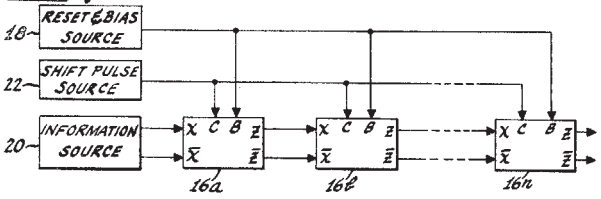
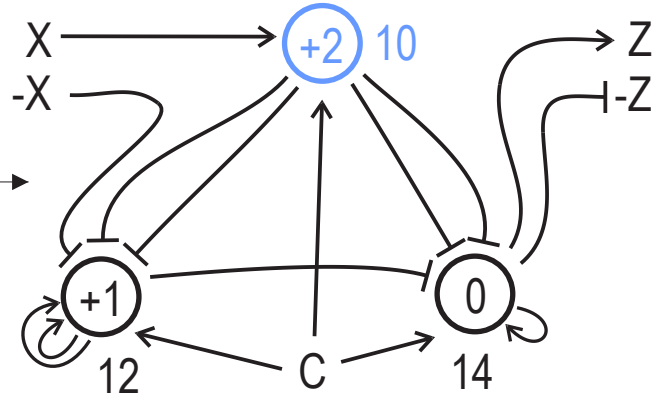
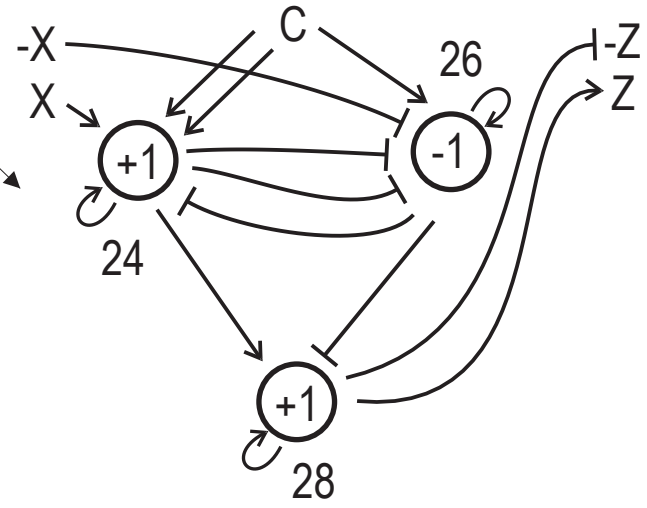


Fig. 3.



Inventor:
KENNETH R. KAPLAN
By
Attorney





Experiment 53
Shift register

We want to finish the digital experiments with a circuit that plays an important role in digital electronics in addition to the counter stages: the shift register. In the beginning, we already had a setup with gene modules where a »1« was shifted in a ring. A shift register, even a simple one, does a little more. It can shift a binary pattern, and not only a single »1« from cell to cell! If we connect, as before, the end of the register with its beginning, we again have a ring shift register. Based on previous knowledge, we would expect that for the setup of a memory cell, at least two »cross-coupled« threshold modules are necessary.

If we set up the register with threshold modules that always sum signals, including the clock signal, we have to solve a general problem: When the shift clock is active, the data content of the cell may be shifted only to the next cell. Since all cells are controlled at the same time by the same clock to take the content of the previous cell, it can easily happen that not only the old content of the previous cell, but already its new content, is taken. Since the previous cell does the same action, the binary pattern could run, in a worst-case condition, in circles as long as the central clock has an active potential. This, of course, has nothing to do with organized shifting.

To put things right, we establish a temporary storage, again consisting of two threshold modules; it is controlled by a second (transfer) clock signal, which is phase-delayed relative to the first one. A memory stage, therefore, contains four threshold modules and operates with two clocks, which requires quite a bit of effort.

But luckily, there are clever people showing that it also works simply with only three modules and one clock. US patent No. 3458734 from 1969 gives two solutions at once that we can implement with our threshold modules (see figure) [23]. Both are set up with threshold modules from the company RCA, having three inputs with weight 1 and two inputs with weight 2 [5].

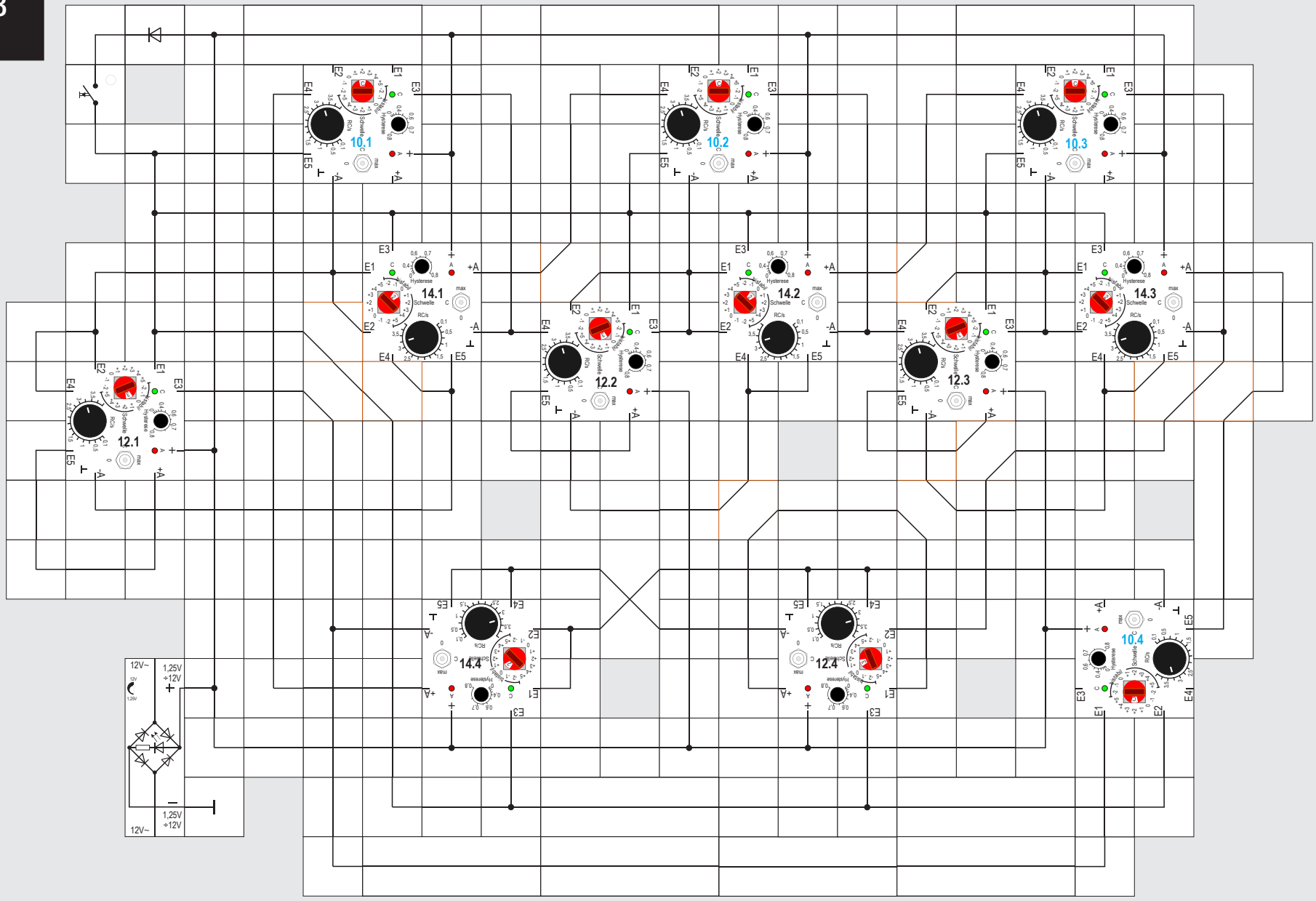
Fig. 1 shows the first implementation. The upper module has the threshold +3, the lower ones have the threshold +4. X and its complement $\overline{X}$ are the inputs of the shift register, Z and $\overline{Z}$ the outputs. The central clock is C, and the common reset signal is B.

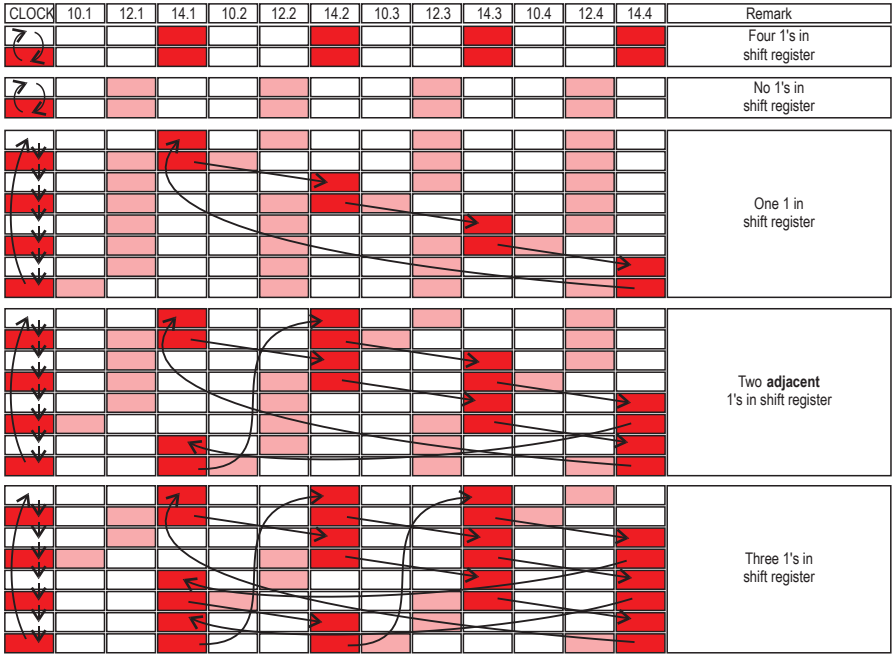
In shift operation, B is 0; with B = 1, all cells should be reset according to the description. But we found that with B=1 all cells are set. Therefore we omit this rather useless function but, as we already know, have to convert it a little, because our modules don't provide the complements $\overline{X}$, $\overline{Y}$ und $\overline{Z}$, but -X, -Y and -Z. We use the equation $X+\overline{X} = 1$ and get $\overline{X} = 1-X$, and analogously for $\overline{Y}$ and $\overline{Z}$. Wherever we need the complement, we take the negative variable and a constant 1-signal. The latter immediately disappears if we decrease the threshold at the same time by 1. We also omit constant 0 signals, like the B signal, which is 0 at the shift operation, without further ado.

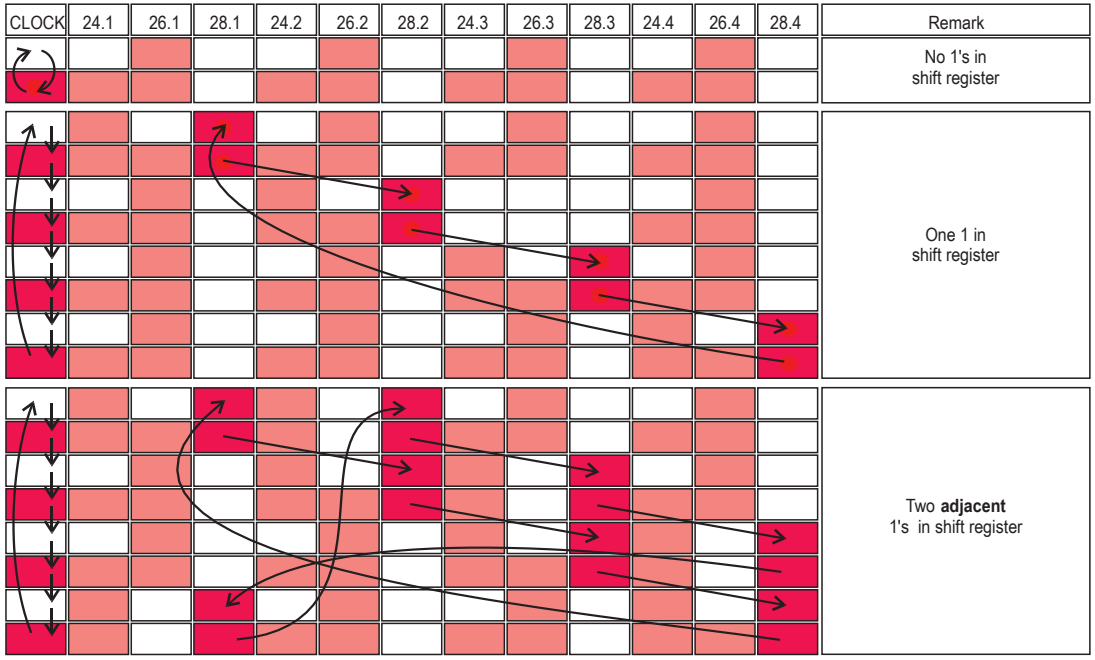
Feedback from outputs to inputs of the same module is realized with a »+1 bistable« adjustment, by which we get an additional free input. That is why Y has to be sent to the input at the lower left module only once externally, although it has the weight +2. After the transformation, we get the circuit in the upper right figure with the changed signals and thresholds.

The central clock is created manually with a button. The diode is used for dropping the supply voltage as usual. The setup shows a ring shift register with four stages, where number 14.n is the module with the actual content of the stage (red in timing chart), and 10.n and 12.n are auxiliary modules that ensure that the content of the stage (pink) is shifted only to the next stage when the clock signal C = 1.

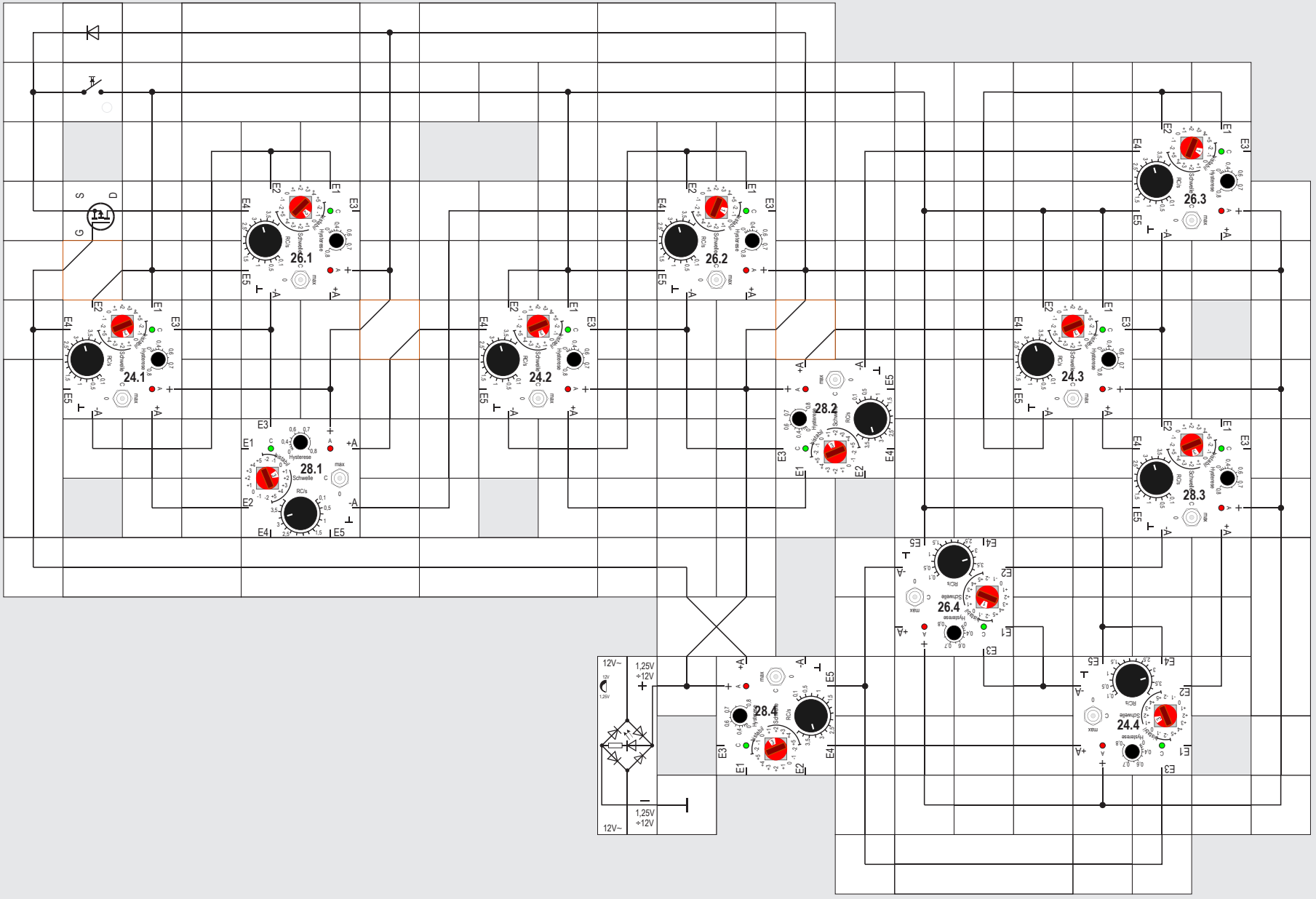
After applying the supply voltage, there are four ones in the register. This is easiest to change manually by using the toggle switch of the Y-register 12.n. This register is at C = 0, always complementary to the Z-register 14.n. The following timing chart shows how the possible contents in the shift register are shifted by the clock.


10.1
10.2
10.3
14.1
14.2
14.3
12.1
12.2
12.3
14.4
12.4
10.4
12V~
1.25V
÷12V
12V~





See page 181 for details of the timing chart, shift register (alternative version)

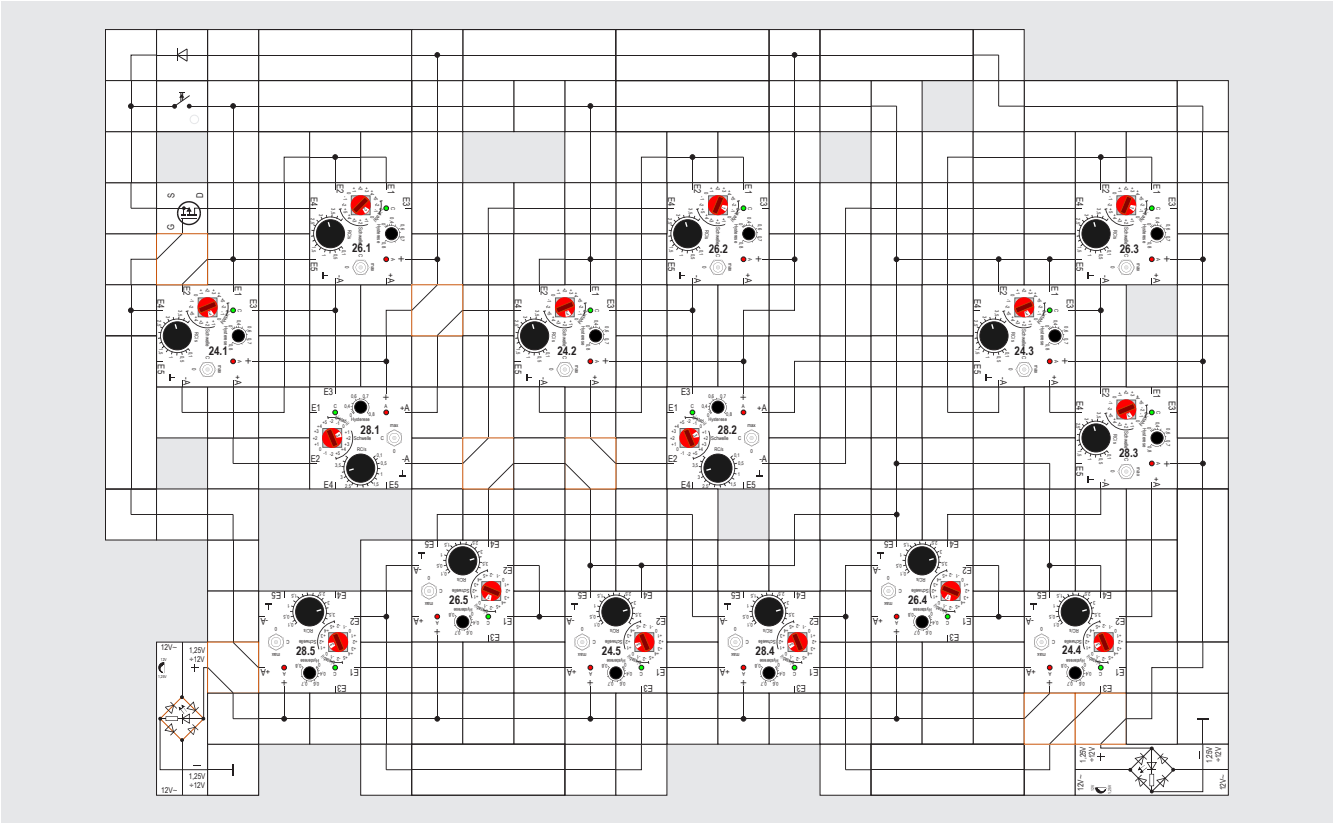

S
D
G
26.1
26.2
26.3
24.1
24.2
24.3
28.1
28.2
28.3
26.4
28.4
24.4
12V~
1,25V
÷12V
12V~
1,25V
÷12V

Experiment 54

Shift register, alternative version

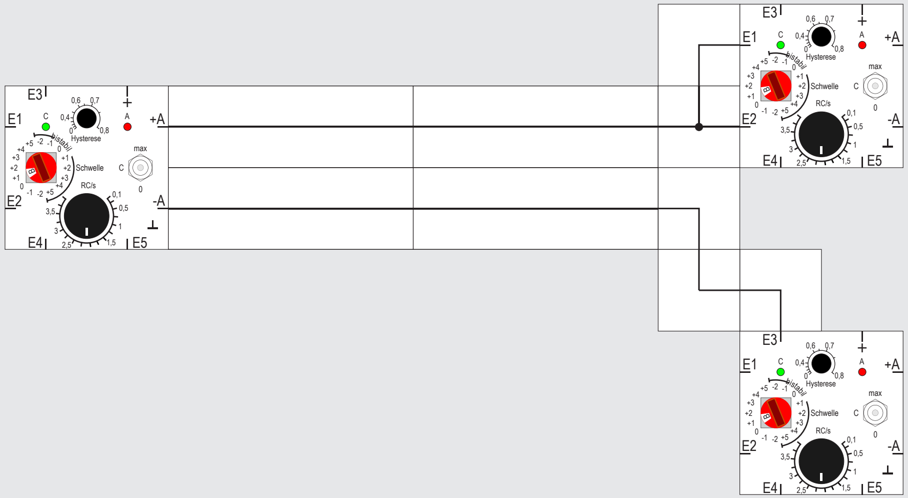
Fig. 2 (page 176) shows a simpler, alternative version of a stage. In this version, the B-signal has to be 1 when shifting, because with B = 0 all stages are reset. We have to omit this function because our threshold modules don't have enough inputs. So B is replaced by two constant 1 signals that disappear with the threshold of the left upper module 24.n decreased by 2. The two upper modules have a threshold of +4 in the original figure, and the lower one has a threshold of +2. The corresponding transformation, as in the first setup, gives us the cell structure on page 176 down at the right edge.

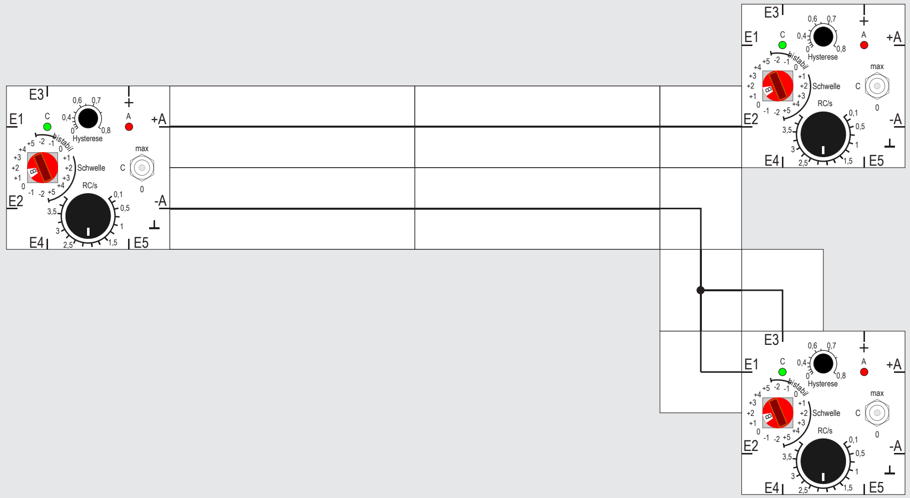
It is easy to build it with our threshold modules. But we again have to use a trick in the setup: We lack space to connect the required -Z4 -signal of module 28.4 to the input of module 26.1. We create the complement -Z4 from $\overline{Z}$ with an inverter and, as compensation, increase the threshold of 26.1 by 1.

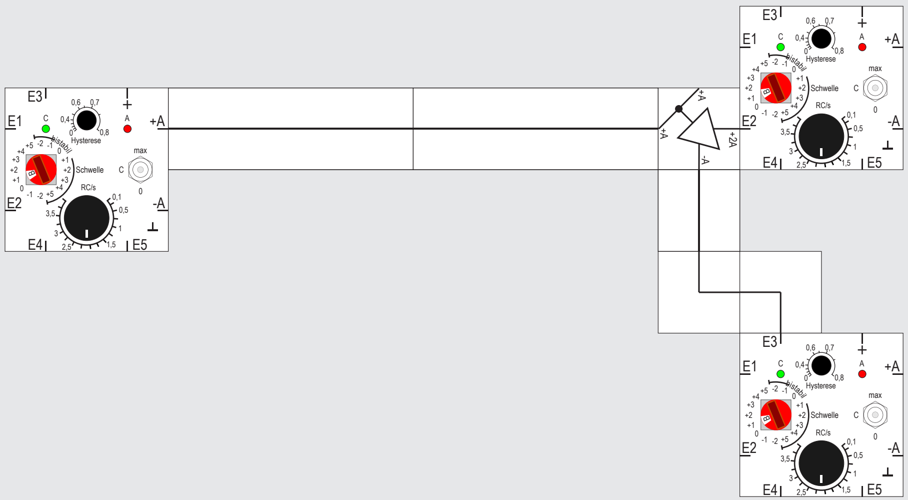


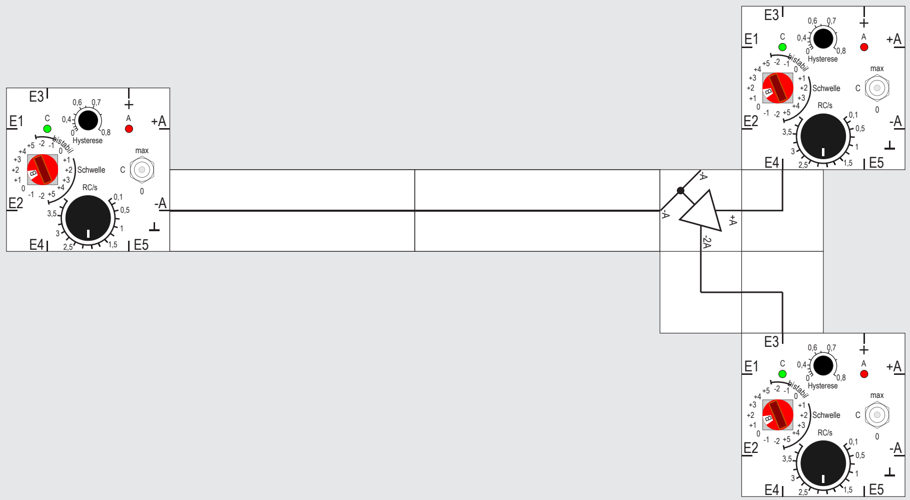
The right diagram on page 179 shows as usual how different contents are shifted by clock control in the register. The modules 26.n and 28.n have complementary contents when the button is released C = 0). When C = 1, all 24.n cells are set (Y = 1).

The assembly board has enough space left to set up a fifth cell. Those who have purchased additional modules and want to build such a register can find a possible setup above. We strongly recommend using a second supply voltage module.


E1
E2
E3
E4
E5
+A
-A
Hysterese
Schwelle
RC/s
max


E1
E2
E3
E4
E5
+A
-A
Hysterese
Schwelle
RC/s
max

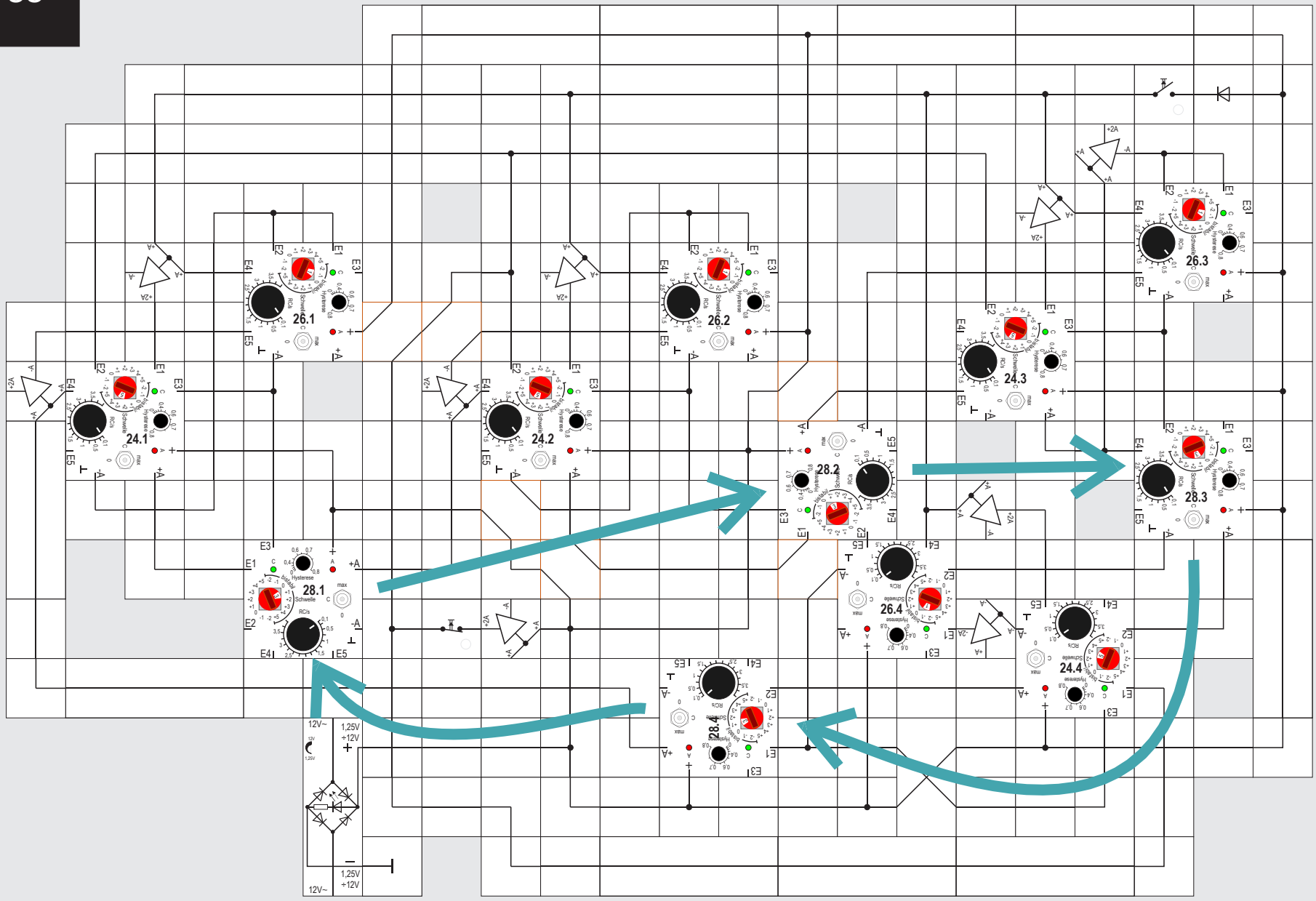

E1
E2
E3
E4
E5
+A
-A
+2A
Hysterese
Schwelle
RC/s
max


E1
E2
E3
E4
E5
+A
-A
-2A
Hysterese
Schwelle
RC/s
max

Useful supplementary modules

There are setups in the biological part as well as in the digital part of our construction kit where we reached limits; either the number of inputs our gene module has was too small or we lacked space on the DIN A2 assembly board, so we could not place all necessary connections for the circuit. We found some tricks that helped.
In the first case, we could change some weights of the input signals by reconsideration or used a diode combination of signals to solve the problems. Lack of space could be overcome sometimes by using a MOSFET inverter to generate the necessary signal only at the spot where it was required. Thus, we saved connection modules and therefore space.
With two subsequently developed ad hoc modules for the gene construction kit, we are now able to solve problems in a very elegant way. Both modules contain a charge pump we introduce on page 160.

As we found out there, the charge pump is able to generate a negative voltage out of a positive one. The less current it has to deliver, the better it does its job. And this is exactly the case in our setups.
If a gene module has to provide different gene modules with both of its output signals, +A and -A, we must lay out two connections in separate ways from its outputs to the receiving modules. This costs a lot of space and modules (see figure above).
Here the »converter« module (order no. 2490) offers relief. We only place the +A signal on the assembly board from the sender to the receiver and invert it at a suitable position. The longer the connection line, the more we can save modules and space. But it gets even better. The charge pump needs a rectangular signal to control the switches, and we can utilize this signal in a so-called Villard circuit to double the voltage. Thus, the converter module not only generates the negative -A voltage from the +A voltage, but also the doubled voltage +2A, precisely $2(+A - U_{th})$ due to the two diodes the circuit is built of. As there are Schottky diodes used, the voltage only drops about 0.4V ($= 2U_{th}$), which the gene modules cope with. If a signal requires the weight 2, we no longer need two inputs of the gene module, only one, and the other is freed for another signal. It might be odd to call the input of the converter module +A, but we always have to bear in mind that this input should only be connected to a +A output of a gene module or to the supply voltage. Moreover, the input is connected to two contacts, so we often save an additional T-connection module.
It would be particularly good if we had such a converter module for a negative voltage, too. That would enlarge our flexibility. This desirable feature can also be fulfilled with the minus converter (order no. 2491). A similar circuit (like the one in the converter module) does its job inside this module. The only difference is that the converter works between the potentials +A and ground, but the minus converter works between ground and -A. Internally there is no difference except the wiring; externally all the potentials are 8V less. Thus, with the input -A (connected to two contacts again), it provides -2A and +A ($-2U_{th}$). We are only allowed to connect -A to a -A output of a gene module; otherwise the module will get damaged. When using both converter modules in a setup, this should be done with reasonable care, as they look very similar (see again figure above).
As an example we will show the setup of the shift register (experiment 54), in which we couldn't realize the central reset function due to lack of signal inputs. But now we can, as you will see on the following pages.
We are quite sure you will use both converters when you reach the previously existing limits in building up new gene networks.


26.1
26.2
26.3
24.1
24.2
24.3
28.1
28.2
28.3
26.4
24.4
28.4
12V~
1,25V
÷12V

Shift register with reset function

In experiment 54 we had to use a trick to connect the outputs of the fourth cell to the inputs of the first and to form in this way a ring-shift register; there was simply a lack of vertical columns on the assembly board. Moreover, we did the setup just from the first module, without a central reset signal, because the gene modules did not have enough inputs. We will now try to substitute the MOSFET inverter and to install the reset function.

Let's start with the MOSFET circuit, which transforms the +A signal from module 28.4 into an $\overline{A}$ input signal for module 26.1. As a matter of fact, this module needs a -A signal, but we compensated for this by adapting the threshold. If we now use the converter module instead of the MOSFET circuit, the +A signal from module 28.4 still reaches input E5 of module 24.1; at the same time we have the converter-generated -A signal for the E4 input of module 26.1. We have to adapt the threshold of the latter to »-1 bistable« as it is at all 26.n modules. That was dead easy, wasn't it? All of the 24.n modules need the central reset signal (called B in the patent application) with a weight of 2. But all inputs are already occupied. First we have to free one input at each of these modules. We can do so as follows: Doubled by a converter, the clock signal (called C) that comes from the push-button module is then connected to only one input of each of the 24.n modules. Since the clock signal C is needed at each 26.n module with weight 1, we do this near each 24.n module. So we require four converter modules. In principle it would be possible to generate the doubled signal centrally by only one converter module and to lead it to the four 24.n modules, but that needs additional space that we unfortunately don't have. We will come back later to the central clock generation, anyway.

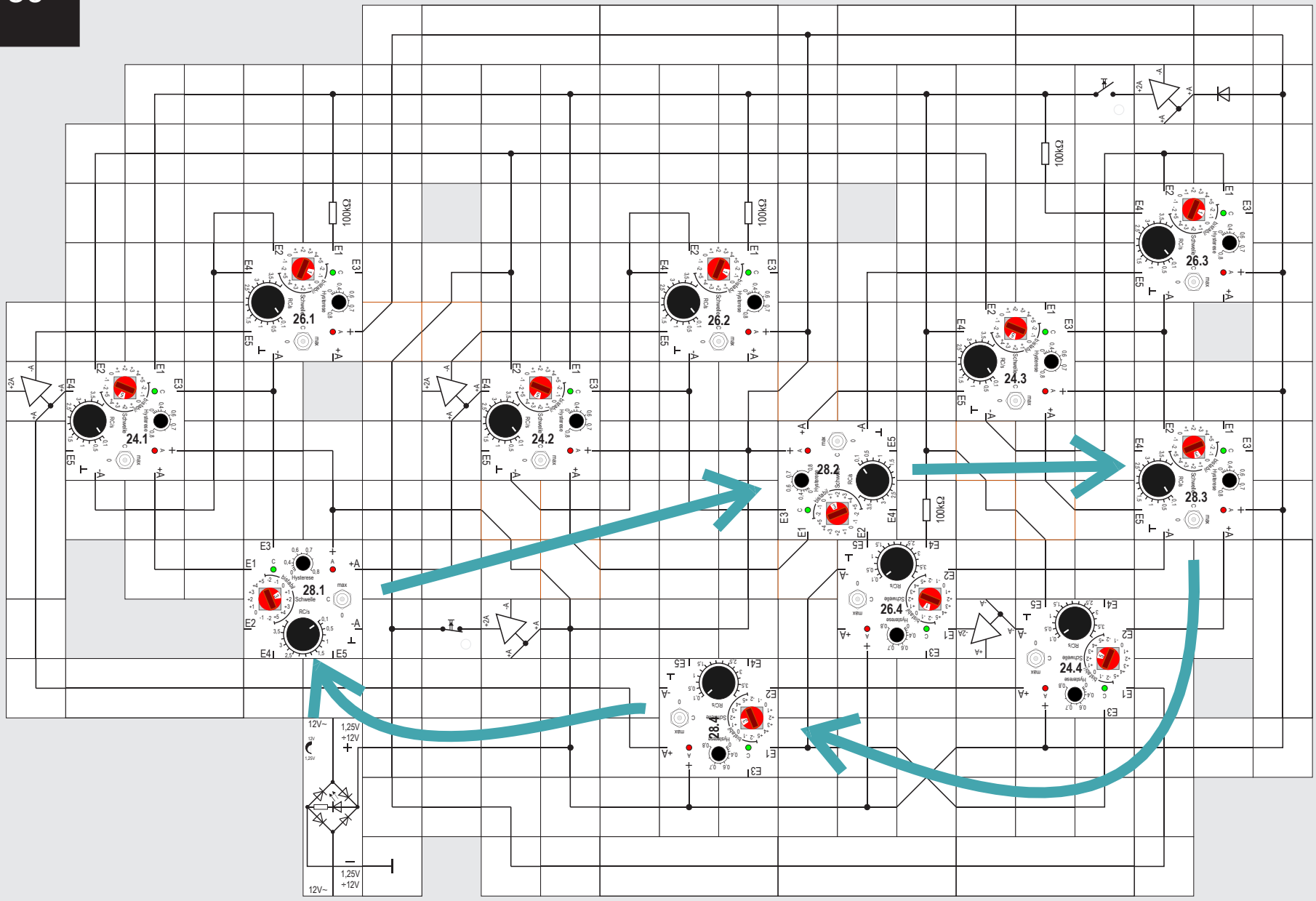
At each 24.n module, we now have a free input, which has to be connected to the reset signal with weight 2.

The reset signal gets active with B = 0. Since we have not used it so far, we could set it mentally to B = 1, e.g., inactive, and by adapting the threshold, simply omit it entirely. We now do a similar thing. We generate the inactive doubled reset signal centrally with a converter out of the supply voltage and connect it to the free inputs of each of the 24.n modules.

We have to increase the thresholds of all these modules from +1b to +3b to guarantee B remains inactive when it equals 1.

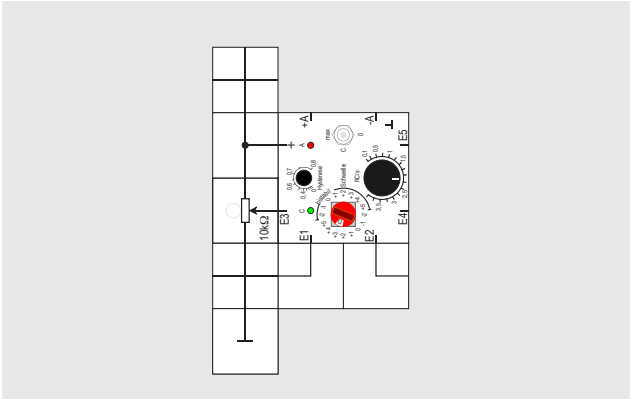
We achieve the active reset with B = 0 by using a push button with normally closed contacts. When we press it, all four 24.n modules no longer get the doubled reset signal, their internal sums don't reach the thresholds, and the cells are reset. Adding the centrally generated reset signal requires space on the assembly board that we don't have at first. Luckily there is enough space on the left-hand side on the bottom of this board; by streamlining the fourth cell, we can use it. More converters are still necessary: One of them has to generate a -A signal for 26.2 from +A signal 28.1, and the other one has to do the same with the +A signal 24.3 to get the needed double -A signal for the 26.3 module. A minus converter is used, too. With its help, the -A signal from 24.4 can be connected twice to module 26.4 in a space-saving way.

The additional blue arrows illustrate which modules store the actual content of the cell (namely the 28.n ones). At the same time they show how the information in the cells is shifted when the button is pressed. All gene modules are set to Hyst = 0 and RC = 0.1s for fast operation. Nevertheless, we should press the clock button for at least half a second to ensure a reliable shift of information.


100kΩ
26.1
26.2
26.3
24.1
24.2
24.3
28.2
28.3
28.1
26.4
24.4
28.4
12V~
1,25V
÷12V

Shift register with reset function, second version

For those who do not want to use that many converter modules, here is the already announced solution: After centrally generating the clock C, its signal is doubled by a converter module and connected to all of the 24.n modules. Since all 26.n modules need it as well, but only with weight 1, we connect each of them via a 100kΩ resistor. Together with the internal 100kΩ input resistance, the input current is halved by this measure and thus the doubling is reversed. We save space on the assembly board, which we can use for linking the power supply lines.
The examples show very well in which ways we can fit additional functions into the setups with the help of the converter modules.



Fractional thresholds

Finally, we would like to state explicitly a feature of the gene modules we avoided in experiment 45. For the time being, we can choose the threshold as an integer only in the range from -2 to +5. If a module has an unconnected input, we can connect it to the center tap of a 10kΩ potentiometer that works between the supply voltage and ground. With this device we are now able to adjust each value continuously, and thus the bias may assume fractional values.
If for example the tap is in the center position, the input gets 4.5V, expressed in threshold values of about +½. To reach a chosen threshold of +3, for example, the sum of the other input signals has to be +2½ only to activate the module.

...since there is a little bit of space left here, you get the derivation of both XOR setups from their Boolean expressions as a bonus. On page 28 (left and right) they appeared out of nowhere. Let's have a look at two variables first.

From $X = A\overline{B} \vee \overline{A}B$

with $\overline{A} = 1 - A$ and $\overline{B} = 1 - B$:

$X = A(1-B) \vee (1-A)B$

$X = (A-AB) \vee (B-AB)$

$X = A \vee B \vee (-2)AB$

we get $X = \langle A + B -2\langle A + B\rangle_{2:1}\rangle_{1:0}$

That was easy, too, wasn't it? And if you calculate with three variables, it's only a bit more complicated:

$$X = A\overline{B}\,\overline{C} \vee \overline{A}B\overline{C} \vee \overline{A}\,\overline{B}C \vee ABC$$

with $\overline{A} = (1-A)$; $\overline{B} = (1-B)$ and $\overline{C} = (1-C)$

$$X = [A(1-B)(1-C)] \vee [(1-A)B(1-C)] \vee [(1-A)(1-B)C] \vee [ABC]$$

$$X = [A \vee ABC \vee (-AB) \vee (-AC)] \vee [B \vee (1 \vee AC) \vee (-AB) \vee (-BC)] \vee [C \vee (1 \vee AB) \vee (-AC) \vee (-BC)] \vee [ABC]$$

$$X = [A \vee (1 \vee BC) \vee (-AB) \vee (-AC)] \vee [B \vee (1 \vee AC) \vee (-AB) \vee (-BC)] \vee [C \vee (1 \vee AB) \vee (-AC) \vee (-BC)] \vee [ABC]$$

(=1) (=1) (=1)

$$X = A \vee B \vee C \vee (-2)(AB \vee AC \vee BC) \vee ABC$$

$$X = A(1 \vee BC) \vee B \vee C \vee (-2)(AB \vee AC \vee BC)$$

(=1) majority function; see page 28

$$X = A \vee B \vee C \vee (-2)(AB \vee AC \vee BC)$$

$$X = \langle A + B + C -2\langle A + B + C\rangle_{2:1}\rangle_{1:0}$$

We get the same result with C = 0 as above.

Components survey

Lectron

	Theme	Aktivation of gene module	Oscillator	Memory cell	Memory cell bistable alternative	Dominant set or reset function	Coincidence memory	Realizable threshold functions	Time constant and hysteresis	2 coupled gene modules (++)	2 coupled gene modules (--)	2 coupled gene modules (+-)	2 master-slave coupled gene modules	2 singly coupled oscillators	2 reciprocally coupled oscillators	Toggle switch	Coupling of three modules (---)	Repressilator	Coupling of three modules (++-)	Coupling of three modules (+--)(+++)	Double coupling of three modules (±±±)	Double coupling of three modules (===)	Coupling of three oscillators	Ring circuit with 4 genes	Ring circuit with 4 genes double coupling	Sum of components to experiment 23
	Exp - no.	1	2	3	4	5	5A	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	
Component	Order no./p.	16	18	20	22	24	25	28	30	32	34	36	38	40	42	44	46	48	50	52	54	56	58	60	62	
Line connection	2103			1	2	2	2	13		2	4	3	1	1	2	2	4	4	2	3	9	7	4	6	13	13
Ground connection	2101	1	1	2	2	3	1	2	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	3
T-connection	2104			2	2	3	3	9		1	1	1	1	1	1	1	2	4	2	2	2	5	3	3	3	9
Angle connection	2105		3	7	4	6	6	7	3	2	2	2	4	7	8	2	3	5	3	3	11	12	10	3	16	16
Double angle connection	2130																									
Crossing non isolated	2106																						1	1		1
Crossing isolated	2107			2	1	1		2		1	1	1	1	1	1	1	1	2	1	1	6	6	1	1	4	6
Triple crossing isolated	2131																									
Triple line connection	2108							1													2	1	2		4	4
Double crossing isolated	2109																									
Resistor 3,9kOhm	2208																									
Resistor 10kOhm	2211			1	1	1		1																		1
Resistor 20kOhm	2212																									
Resistor 47kOhm	2213																	1								1
Resistor 100kOhm	2214																									
Potentiometer 10kOhm	2219																									
Potentiometer 250kOhm	2221																									
Resistor 470kOhm	2230																									
Elektrolytkond. 100µF	2305																									
RC-circuit 10kOhm/10µF	2316																									
Photoresistor	2216																									
Diode	2402			1	1	2	1	1																		2
Schottky diode	2406																									
p-channel MOSFET	2444																									
LED superbright	2474																	1								1
Op-amp	2476																									
LED display	2477																									
Timer 555	2480																									
Gene module	2482	1	1	1	1	1	1	2	1	2	2	2	2	2	2	2	3	4	3	3	3	3	3	4	4	4
Converter	2490																									
Minusconverter	2491																									
normally off press button	2501			2	2	2	2	3																		3
norm. closed press button	2502						1									2										2
Power supply module	2601	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Assembly board DIN A3	2961	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Assembly board DIN A2	2960																									
Sum		3	6	20	17	22	18	42	6	10	12	11	11	14	16	12	15	24	13	14	35	36	26	20	46	46

	Theme	Ring circuit with 8 genes double coupling	Feed-forward loops	Coupled oscillators	Simulation model of bakers' yeast	Light-dependent division of baker's yeast	Fine adjustment of the delay	Simulation model of fission yeast	Simulation of Arabidopsis	Signal combination by diodes	Clock generator with timer	Clock generator with timer (alternative)	Switching LUG automatically	Turning LFY automatically	Switching SUP automatically	Simulation of Arabidopsis without a timer	Sim. segment development red flour beetle larva	Sim. segment development red flour beetle grub	Sim. segment development Drosophila	Cyclic simulation of *Drosophila melagonaster*	Sim. of differentiation hematopoietic stem cells	Differentiation of stem cells	Cyclic simulation of stem cells	Forming of swarms	Quorum sensing	Improved model I for quorum sensing	Improved model II for quorum sensing	Repressilator and quorum sensing	Reppressilator & quorum sensing (alternative)	Counter with 4 gene modules	Counter with 4 gene modules (alternative)	Counter with 6 gene modules	Counter stage	Extending the counter	Counter stage (alternative setup 1)	Counter stage (alternative setup 2)	Shift register	Shift register (alternative)	Sum of all components		Shift register with reset function	Shift register with reset function (alternative)
	Exp - no.	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	39A	40	40A	41	42	42A	43	44	45	46	47	48	50	51	52	52A	52B	52C	52D	53	54			55	56
Component	Order no./p.	64	66	68	72	82	84	86	92	101	104	106	107	107	110	114	118	122	124	130	132	138	142	148	150	152	154	156	158	164	166	168	170	172	174	175	176	180			184	186
Line connection	2103	33	18	20	21	22	12	24	20	43	1	46	47	45	49	49	23	20	38	34	23	47	47	48	47	47	49	48	32	14	12	20	31	42	43	45	49	48	49		61	69
Ground connection	2101	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	4	4	1	1	1	1	1	2	2	1	1	1	4		1	1
T-connection	2104	7	8	8	23	23	13	19	14	29	6	34	34	34	34	33	18	20	21	21	21	25	22	21	23	22	27	28	17	9	10	17	19	28	27	25	33	25	34		26	28
Angle connection	2105	28	8	8	25	21	42	25	24	31	4	36	39	42	47	44	25	25	29	31	28	28	36	27	23	24	30	28	30	16	17	24	18	23	23	25	47	43	47		38	40
Double angle connection	2130				5	7		8	6	9		9	9	10	10	5	6	4	5	5	9	8	10	3	7	5	8	3	6								10	3	10		6	8
Crossing non isolated	2106				2	2		1	5	1		2	2	2	3	2	3	1	3	3	4	1	3				2			1	1		3	4	3	5	5	4	5		2	2
Crossing isolated	2107	8	2	2	32	32		18	18	30	1	30	31	33	34	30	11	11	21	32	20	31	33	21	20	16	22	12	9	10	10	14	19	20	23	23	27	23	34		35	37
Triple crossing isolated	2131				6	6		3		6		6	6	6	6	7	4	4	8	10	3	10	18	4	2	3	1	1	1			2	2	6	4	4	10	4	18		4	3
Triple line connection	2108	8	4	6	6	5	1	7	4	6		7	10	10	10	8	5	4	3	3	4	11	9	7	4	5	8	11	11	8	8	6			1		11	11	11		14	10
Double crossing isolated	2109								1	1		1	1	1		1			2	2							1					1					1	1	2		1	1
Resistor 3,9kOhm	2208																										1	1											1			
Resistor 10kOhm	2211										2																2	2						1	1				2			
Resistor 20kOhm	2212											1	1	1	1																								1			
Resistor 47kOhm	2213																																						1			
Resistor 100kOhm	2214										1	2	2	2	2												4	4											4			4
Potentiometer 10kOhm	2219																									1	1	1											1			
Potentiometer 250kOhm	2221																										1	1											1			
Resistor 470kOhm	2230										1	1	1	1	1																								1			
Elektrolytkond. 100µF	2305										1	1	1	1	1												1	1											1			
RC-circuit 10kOhm/10µF	2316																									4													4			
Photoresistor	2216					1																																	1			
Diode	2402																													1		1		1	2		1	1	2		1	1
Schottky diode	2406									2	1	3	3	7	7	6							2			4													7			
p-channel MOSFET	2444																																	1	1		1	1	1			
LED superbright	2474																																						1			
Op-amp	2476																										2	2											2			
LED display	2477																										1	1											1			
Timer 555	2480										1	1	1	1	1																								1			
Gene module	2482	8	6	6	12	12	13	10	12	11		11	11	12	12	13	11	10	13	13	11	8	8	12	13	13	12	12	13	4	5	6	9	12	12	13	12	12	13		12	12
Converter	2490																																								8	4
Minusconverter	2491																																								1	1
normally off press button	2501																													1		1		1	1		1	1	3		1	1
norm. closed press button	2502						1																																2		1	1
Power supply module	2601	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		1	1
Assembly board DIN A3	2961	1	1	1			1				1							1												1	1	1							1			
Assembly board DIN A2	2960				1	1		1	1	1		1	1	1	1	1	1		1	1	1	1	1	1	1	1	1	1	1				1	1	1	1	1	1	1		1	1
Sum		94	48	52	134	133	85	117	106	171	22	193	201	210	220	200	108	101	145	156	125	171	190	145	141	147	178	161	121	67	65	95	103	142	144	142	210	179	266		213	224

Does not include experiments 4A and 49; they are not necessarily essential for the gene regulation theme; components for experiments 55 and 56 for information only.

References

[1] E. Schrödinger: What Is Life? The Physical Aspect of the Living Cell. (1944) Cambridge University Press, Cambridge.

[2] B. Alberts et al. Molecular Biology of the Cell. (2014) Garland Science, New York.

[3] S. Bornholdt. Boolean Network Models of Cellular Regulation: Prospects and Limitations, J. Roy. Soc. Interface 5 (2008) S85-S94.

[4] S. Braunewell and S. Bornholdt, Superstability of the Yeast Cell-Cycle Dynamics: Ensuring Causality in the Presence of Biochemical Stochasticity, J. Theor. Biol. 245 (2007) 638-643.

[5] S.L. Hurst. Threshold Logic: An Engineering Survey (1971) Mills & Boon, London.

[6] M.L. Dertouzos. Threshold Logic: A Synthesis Approach (1965) MIT Press, Cambridge, MA, USA.

[7] R.O Winder. Threshold Functions Through n = 7. No. SR-7. RCA Labs, Princeton, NJ, 1964.

[8] T.S. Gardner, C.R. Cantor, J.J. Collins. Construction of a Genetic Toggle Switch in *Escherichia coli*. Nature 403 (2000) 339-342.

[9] M. Elowitz, S. Leibler. A Synthetic Oscillatory Network of Transcriptional Regulators. Nature 403 (2000) 335. See also: http://en.wikipedia.org/wiki/Repressilator

[10] S.S. Shen-Orr, R. Milo, S. Mangan, U. Alon. Network Motifs in the Transcriptional Regulation Network of Escherichia coli. Nature Genetics 31 (2002) 64-68.

[11] T. Danino, O. Mondragón-Palomino, L. Tsimring, J. Hasty. A Synchronized Quorum of Genetic Clocks. Nature 463 (2010) 326. See also:

http://2013.igem.org/Team:KU_Leuven/Project/Oscillator/Design

[12] F. Li, T. Long, Y. Lu, Q. Ouyang, C. Tang. The Yeast Cell-Cycle Network Is Robustly Designed. Proc. Natl. Acad. Sci. USA 101 (2004) 4781.

[13] M.I. Davidich and S. Bornholdt, Boolean Network Model Predicts Cell Cycle Sequence of Fission Yeast PLoS ONE 3 (2008) e1672.

[14] M.I. Davidich. S. Bornholdt. Boolean Network Model Predicts Knockout Mutant Phenotypes of Fission Yeast. PLoS ONE 8 (2013) e71786.

[15] L. Mendoza, D. Thieffry, E.R. Alvarez-Buylla. Genetic Control of Flower Morphogenesis in *Arabidopsis thaliana*: A Logical Analysis., Bioinformatics 15 (1999) 593.

[16] C.P. Choe, S.C. Miller, S.J. Brown. A Pair-Rule Gene Circuit Defines Segments Sequentially in the Short-Germ Insect *Tribolium castaneum*. Proc. Natl. Acad. Sci. USA 103 (2006) 6560. http://www.pnas.org/content/103/17/6560.full

[17] S.B. Carroll, J.K. Grenier, S.D. Weatherbee. From DNA to Diversity: Molecular Genetics and the Evolution of Animal Design. Blackwell Publishing: Cambridge, MA (2001) p. 59, Fig. 3.5.

[18] J. Krumsiek, C. Marr, T. Schroeder, F.J. Theis. Hierarchical Differentiation of Myeloid Progenitors Is Encoded in the Transcription Factor Network. PLoS ONE 6 (2011) e22649.

[19] B.D. MacArthur, C.P. Please, R.O.C. Oreffo. Stochasticity and the Molecular Mechanisms of Induced Pluripotency. PLoS ONE 3 (2008) e3086.

[20] J. Garcia-Ojalvo, M.B. Elowitz, S.H. Strogatz. Modeling a Synthetic Multicellular Clock: Repressilators Coupled by Quorum Sensing, Proc. Natl. Acad. Sci. USA 101 (2004) 10955.

[21] E.H. Hellen, S.K. Dana, B. Zhurov, E. Volkov. Electronic Implementation of a Repressilator with Quorum Sensing Feedback, PLoS ONE 8 (2013) e62997.

[22] P. Misiurewicz. Comment on: Counting with Majority-Logic Networks. IEEE Transactions on Electronic Computers 14 (1965) 262.

[23] K.R. Kaplan. Shift Registers Employing Threshold Gates (1969) US Patent No. 3458734.

Gene Regulation: Experiments
with the LECTRON Experimental Kit

ISBN 978-3-00-051441-8

Published 2016 by
Lectron GmbH
Reha Werkstatt Oberrad
Buchrainstr. 18
D-60599 Frankfurt
Tel.: +49 69 90 50 12 82
Fax: +49 69 90 50 12 83
www.lectron.de