Algebraic and Combinatorial Algorithms for Translinear Network Synthesis

David Ilsen, Ernst Josef Roebbers, Gert-Martin Greuel

***Abstract*— We propose a new approach for static translinear network synthesis. We model the network topology in terms of graph theory, leading to a catalog of valid translinear networks. The catalog serves as a synthesis tool for circuits with given desired input-output behavior. Methods from algebraic geometry and computer algebra are used to match the desired behavior with the models from the catalog, turning our approach into a powerful synthesis tool for analog circuits. The strategy is applied successfully to new circuits for industrial use.**

***Index Terms*— translinear circuits, circuit design, circuit topology, cataloging of topologies, computer algebra**

CONTENTS

D. Ilsen is with zeb/Information.Technology GmbH & Co KG, Münster
E. J. Roebbers is with Analog Microelectronics GmbH, Mainz
G.-M. Greuel is with the University of Kaiserslautern

I. INTRODUCTION

In this paper we develop a mathematical model of a special class of analog microelectronic circuits called *translinear circuits.* The goal is to provide foundations of a new coherent synthesis approach for this class of circuits, which, at the same time, allows to design algorithms and implementations for practical applications. The mathematical methods of the suggested synthesis approach come from graph theory, combinatorics, algebraic geometry, and, in particular, from symbolic methods from computer algebra. The theory and algorithms presented in this paper have been developed by the first author in his PhD thesis [11]. The algorithms have been implemented using C++, SINGULAR [8], and MATHEMATICA, providing as output netlists in PSPICE format.

Translinear circuits [4], [6] form a very special class of analog circuits. They rely on nonlinear device models, but still allow a very structured approach to network[1] analysis and synthesis. Thus, translinear circuits play the role of a bridge between the "unknown space" of nonlinear circuit theory and the very well exploited domain of linear circuit theory.
The nonlinear equations describing the behavior of translinear circuits possess a strong algebraic structure that is nonetheless flexible enough for a wide range of nonlinear functionality. Furthermore, translinear circuits offer several technical advantages like high functional density, low supply voltage and insensitivity to temperature.
This unique profile is the reason that translinear networks are considered as the key to systematic synthesis methods for nonlinear circuits.

We propose the usage of a computer-generated catalog of translinear network topologies as a synthesis tool. The idea to compile such a catalog has grown from the observation that on the one hand, the topology of a translinear network must satisfy strong constraints which severely limit the number of "admissible" topologies, in particular for networks with few transistors, and, on the other hand, the topology of a translinear network already fixes its essential behavior, at least for static networks, because the so-called *translinear principle* requires the continuous parameters of all transistors to be the same.
Even though the admissible topologies are heavily restricted, it is of course a highly nontrivial task to compile such a catalog. Combinatorial techniques have been adapted to undertake this task [11].
A synthetic catalog of network topologies has been used by circuit designers in a different context: Lists of VCCS topologies for CMOS circuits have been compiled by E. Klumperink,

[1] In this paper, a "circuit" means electronics hardware, wheres a "network" means its mathematical model.

J. Schmitz and others [14], [15], [19], [20]. However, this article provides the first approach to translinear network synthesis. Moreover, it does not only lay the theoretical foundations but the proposed algorithms have been implemented and proven effective in an industrial design process.

In a catalog of translinear network topologies, symbolic prototype network equations can be stored along with each topology. When a circuit with a specified behavior is to be designed, one can search the catalog for a network whose input-output behavior can be matched with the desired behavior.

In this context, two algebraic problems arise: First, to set up a meaningful equation for a network in the catalog, an elimination of internal variables must be performed. The result is one polynomial describing the input-output behavior. Second, to test whether this polynomial equation and a specified equation of desired behavior can be "matched", a system of polynomial equations must be solved, where the solutions are restricted to a finite set of integers. Sophisticated algorithms from computer algebra are applied in both cases to perform the symbolic computations, for which we used the computer algebra system SINGULAR [8].

All mentioned algorithmic methods have been implemented and successfully applied to actual design problems at Analog Microelectronics GmbH (AMG), Mainz.

This paper is organized as follows:

Section II gives a review of the so-called translinear principle, the functional principle of translinear circuits.

Section III then gives an analysis of their topology and develops some abstract notions in order to model the topology in terms of graph theory. This modelling is the heart of the new synthesis approach, since it gives the exact specification of the catalog.

Section IV gives an overview on the data stored in the catalog.

Section V is about the structure of the equations describing a translinear network's behavior, and the first algebraic problem mentioned above, namely the elimination of collector currents from these equations to produce a compact input-output description of the behavior.

The second algebraic problem and some other issues concerning the search in the catalog for a network with a particular behavior are discussed in Section VI.

Section VII reports about the successful application of the developed synthesis methodology in the design of a new humidity sensor system of AMG. Finally we mention some open problems and make suggestions for further research.

Acknowledgements

This research was supported by the DFG-Graduiertenkolleg "Mathematik und Praxis" in Kaiserslautern.

The authors wish to thank the company AMG for the collaboration and Dr. Wouter Serdijn for helpful discussions.

II. THE TRANSLINEAR PRINCIPLE

This section gives a review of the so-called translinear principle, the functional principle of translinear circuits. It has been formulated and given its name by Barrie Gilbert in 1975 [4].

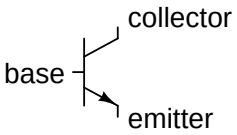


Fig. 1. The symbol for a bipolar NPN transistor, our placeholder for a "translinear device".

A. Translinear Device Model

The translinear principle relies on an exponential voltage-to-current relationship of certain micro-electronic devices. The original "translinear device" is the bipolar NPN transistor, other devices with valid exponential models are diodes, PNP transistors and MOS transistors operating in weak inversion [24]. Recently, an emulation of a bipolar transistor has been proposed [2], where a subnetwork structure of three CMOS transistors and one diode shows the necessary exponential behavior.

In our circuit diagrams, we will use the symbol of a bipolar NPN transistor, shown in Figure 1, to represent an abstract "translinear device", and we will simply use the term "transistor" for such an abstract device. It follows from the above that several different silicon implementations of a "transistor" are possible.

The ideal exponential model of a transistor is given by the equation

$$I_{CE} = I_S e^{V_{BE}/U_T}, \tag{1}$$

saying that its collector current I_{CE} (the current from collector to emitter) is exponentially dependent on the base voltage V_{BE} (the voltage between base and emitter). In this model, I_S and U_T are device- and operation-dependent parameters called *saturation current* and *thermal voltage*, respectively. It is assumed that $I_S > 0$ and $U_T > 0$.

We usually make the additional model assumption that the base current (the current from base to emitter) of a transistor is zero.

B. Translinear Loops

The key structures of translinear networks are so-called **translinear loops**. We call a loop W of the network digraph a translinear loop if it satisfies the following three properties:

- W consists exclusively of base-emitter branches of transistors.
- All transistors involved share the same pair (I_S, U_T) of parameters.
- W consists of as many forward branches as backward branches. (Remember that we regard the branches to point "from base to emitter".)

Figure 2 shows two examples for a translinear loop.

The interesting property of translinear loops is that, due to the exponential transistor model, we can deduce a multiplicative relation of collector currents from KIRCHHOFF's Voltage

Law (KVL): Denote the base voltages of the transistors in W-forward orientation by $V_1^f, \ldots, V_r^f$, the base voltages of the transistors in W-backward orientation by $V_1^b, \ldots, V_r^b$. Then KVL for W reads

$$V_1^f + \cdots + V_r^f = V_1^b + \cdots + V_r^b.$$

Taking advantage of the common parameters, we deduce

$$e^{V_1^f/U_T} \cdot \ldots \cdot e^{V_r^f/U_T} = e^{V_1^b/U_T} \cdot \ldots \cdot e^{V_r^b/U_T}$$

and multiplication by I_S^r yields

$$\left(I_S e^{V_1^f/U_T}\right) \cdot \ldots \cdot \left(I_S e^{V_r^f/U_T}\right) = \\ = \left(I_S e^{V_1^b/U_T}\right) \cdot \ldots \cdot \left(I_S e^{V_r^b/U_T}\right).$$

Considering the model equation (1), this means exactly

$$I_1^f \cdot \ldots \cdot I_r^f = I_1^b \cdot \ldots \cdot I_r^b, \tag{2}$$

where $I_1^f, \ldots, I_r^f$ and $I_1^b, \ldots, I_r^b$ denote the collector currents of the transistors whose base-emitter branches are W-forward or W-backward, respectively.

Remark 1: Note that I_S and U_T don't occur any more in eqn. (2). This means that the relation between the collector currents holds independently of these parameters, provided they are indeed common. One nice effect of this is that translinear networks are essentially temperature-insensitive.

Example 1: The loop indicated by thick lines in Figure 3 is a translinear one, being made up of the base-emitter branches of transistors $Q_1, \ldots, Q_4$. (We assume that U_T and I_S coincide for the four transistors.) Application of the translinear principle yields[2]

$$I_1 \cdot I_3 = I_2 \cdot I_4. \tag{3}$$

[2]Here we simply denote the collector current of a transistor Q_j by I_j. We will stick to this convention in the following examples, too.

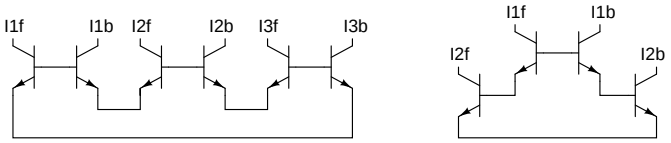


Fig. 2. Examples for translinear loops.

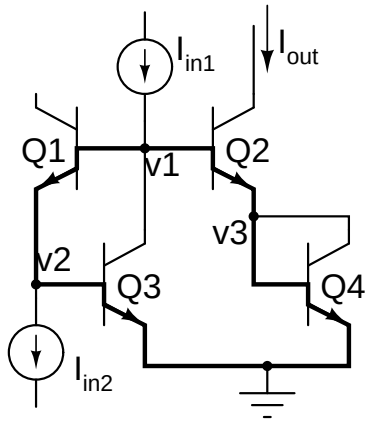


Fig. 3. A geometric mean circuit

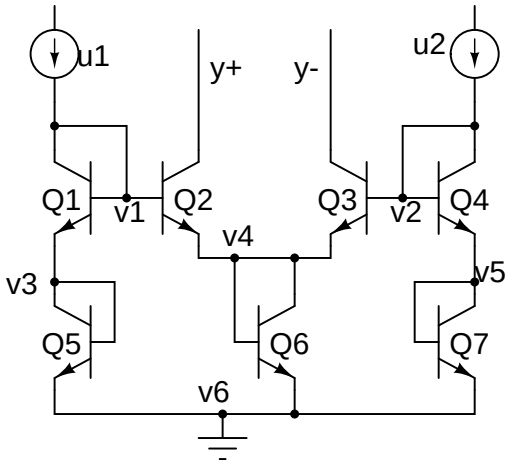


Fig. 4. A translinear frequency doubling network. (Since its first publication by GENIN and KONN in 1979 [3], this network has become a very prominent example application of the translinear principle.)

Now remember our assumption that base currents are zero. Taking this into account, Kirchhoff's Current Law for v_1 means that $I_3 = I_{\text{in1}}$. Similarly for $v_2 : I_1 = I_{\text{in2}}$, and for v_3: $I_4 = I_2 = I_{\text{out}}$. Thus we can substitute the collector currents in eqn. (3) by I_{in1}, I_{in2} and I_{out}:

$$I_{\text{in2}} \cdot I_{\text{in1}} = I_{\text{out}} \cdot I_{\text{out}},$$

so $I_{\text{out}} = \sqrt{I_{\text{in1}} \cdot I_{\text{in2}}}$ (since I_{out}, being a collector current, must be positive). That means, the network "computes" the geometric mean of the two inputs.

Example 2: As an example for a network with several translinear loops, consider the network of Figure 4.

Transistors Q_1, Q_2, Q_6, Q_5 form a translinear loop. The according equation is

$$I_1 \cdot I_5 = I_2 \cdot I_6. \tag{4}$$

Another translinear loop consists of transistors Q_3, Q_4, Q_7, Q_6. This gives

$$I_3 \cdot I_6 = I_4 \cdot I_7. \tag{5}$$

By Kirchhoff's Current Law and our neglection of base currents, we can rewrite eqn. (4) and eqn. (5) as

$$\begin{aligned} u_1 \cdot u_1 &= y^+ \cdot (y^+ + y^-), \\ y^- \cdot (y^+ + y^-) &= u_2 \cdot u_2. \end{aligned}$$

A little computation reveals that

$$y^+ - y^- = \frac{u_2^2 - u_1^2}{\sqrt{u_1^2 + u_2^2}}.$$

If we apply sinusoidal inputs with a 90° phase shift, like

$$\begin{aligned} u_1 &= |a \sin t|, \\ u_2 &= |a \cos t|, \end{aligned}$$

with a fixed $a \in \mathbb{R}$, then the differential output becomes

$$y^+ - y^- = a \cos 2t,$$

that is, the network shows a frequency doubling behavior for these inputs.

C. Motivation for a Catalog of Topologies

The following properties of static translinear (STL) networks can be observed from the examples of the preceding subsection:

- STL networks can be described in terms of currents by systems of polynomial equations, consisting of the translinear loop equations (2) and the node equations.
- In such a system, no continuous parameters occur. This is due to the fact that U_T and I_S vanish from the equations as soon as the STL principle is applied.
- The topology of translinear networks satisfies strong constraints. One of these constraints is the condition that the number of forward and backward branches in a translinear loop must be the same. Another constraint concerns the connection of collectors; it will be considered in Subsection III-B.

The second property means that the behavior of a STL network is already fixed by the network topology. In particular, there is only a finite number of different STL networks when the number of transistors is bounded!

Together with the third property, which says that the number is not only finite but also "not too large", this observation led to the idea of computing a complete catalog of "small" STL networks. If along with each network appropriate equations are stored, such a catalog can serve as a design tool in an obvious way: When the designer is in search for a circuit with a given desired behavior, she or he can simply run through the catalog to find a network whose equations match that behavior.

III. The Topology of Translinear Networks

In this section, we develop a rigorous theory of translinear networks in terms of graph theory. It will be the basic mathematical model of a translinear circuit's topology and consists essentially of a strict formulation of the constraints which the topology of translinear networks has to obey. Although these constraints have been known before (their identification is due to E. Seevinck [21]), their translation into mathematics is new.

The precise mathematical formulation is necessary for the specification of the combinatorial task of compiling complete lists of topologies, which make up the desired catalog.

A. Translinear Digraphs

Translinear directed graphs, or translinear digraphs, are the mathematical objects that are used to represent the core structure of a translinear network, the structure consisting of the translinear loops. For the standard notions from graph theory we refer to [13].

Definition 1: A **translinear digraph** is a directed multigraph G satisfying the following properties:

1) Every loop of G has as many forward branches as backward branches.
2) G is biconnected, i. e. G is connected and remains so after the removal of any node.

One should think of a translinear digraph G as the digraph formed by the base-emitter branches of a translinear network, that is, the branches of G are in 1-to-1-correspondence with the transistors of the network and for each branch e, $\mathrm{tail}(e)$ corresponds to the base node and $\mathrm{head}(e)$ to the emitter node of the respective transistor.

See Figure 5 for an example of a translinear digraph.

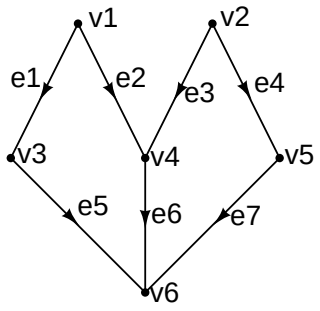


Fig. 5. The translinear digraph of the frequency doubling network of Figure 4. Branch e_j of the digraphs corrsponds to transistor Q_j in Figure 4.

Condition 1 in Definition 1 obviously reflects the main requirement on translinear loops as presented in Section II. The reason for including Condition 2 into the definition is that the loops of different biconnected components of the base-emitter digraph are decoupled, so we can consider the corresponding sub-networks seperately.

The concept of a translinear digraph as the core of a translinear network was introduced by E. Seevinck [21], although he concentrated on undirected graphs.[3] (Seevinck's definition differs from the one given here in some more respects.)

It turns out that condition 1 of Definition 1 has a nice reformulation, as expressed by the following theorem:

Theorem 1: Let G be a digraph. The following two statements are equivalent:

1) Every loop of G has as many forward arcs as backward arcs.
2) There exists a map $r : V(G) \to \mathbb{Z}$ such that

$$\forall\ e \in E(G): \quad r(\mathrm{tail}(e)) = r(\mathrm{head}(e)) + 1.$$

(Here and in the following, $V(G)$ resp. $E(G)$ denotes the set of vertices resp. edges of G.)

The proof of Theorem 1 is not complicated, see [11] for the details.

It is clear that if a map r as in the second statement of Theorem 1 exists, so does a map r_0 that fulfills the same condition as well as the additional property

$$\min_{v \in V(G)} r_0(v) = 0. \tag{6}$$

(Simply define $r_0(v) := r(v) - \min_{v' \in V(G)} r(v')$.) The nodes can then be partitioned into "levels" or "layers", such that a branch always points from one layer to the next lower layer:

$$V = V_0 \,\dot\cup\, V_1 \,\dot\cup \ldots \dot\cup\, V_R,$$

where $V_j = \{v \in V \,|\, r_0(v) = j\}$ and $R := \max_{v \in V} r_0(v)$. This is illustrated in Figure 6.

[3]This is the reason that the term "translinear graph" is found very often in the literature, whereas "translinear digraphs" have not found much attention hitherto.

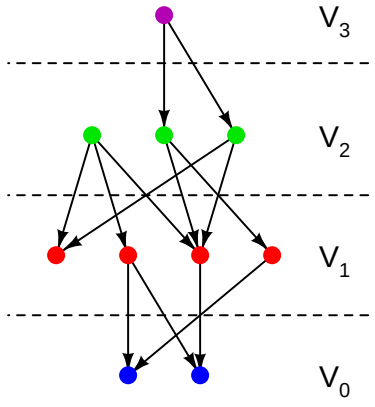


Fig. 6. layers of a translinear digraph

Example 3: For the frequency doubling network (Figures 4 and 5), $R = 2$, $V_0 = \{v_6\}$, $V_1 = \{v_3, v_4, v_5\}$ and $V_2 = \{v_1, v_2\}$.
For a connected digraph, the additional property (6) makes r_0 unique.

Definition 2: Let G be a connected layered digraph. For a node $v \in V(G)$, we call the integer $r_0(v)$ the **rank** of v, and also denote it by $\mathrm{rank}(v)$.
In other words, the rank of a node is the index of the layer it belongs to.

What is crucial about the layers of the translinear digraph of a network is that the rank of a node corresponds nicely to a certain range of the electric potential the node will have in an actual circuit. This is because the voltage drop along a base-emitter branch is always much larger than the swing of potential at one particular node.

Since the rank is a node invariant, it also helps a lot in the classification of translinear digraphs, as will become apparent in Section IV.

B. Connection of Collectors

The previous subsection dealt with the base-emitter connectivity of a translinear network, which can be encoded in a translinear digraph. The main piece of information that is furthermore needed to describe a complete network is where to connect the collectors to.

Let e be a branch of a translinear digraph G. The collector of the transistor corresponding to e can only be connected to a node v of G if $\mathrm{rank}(v) > \mathrm{rank}(\mathrm{head}(e))$. The reason is that $\mathrm{head}(e)$ is the emitter node, and the collector needs a higher potential than the emitter. E. Seevinck considered this condition in the construction of his "T-matrices" [21].

We denote the collector node of a transistor corresponding to a branch e of the translinear digraph by $C(e)$.

Example 4: For the frequency doubling network (Figure 4), we have $C(e_1) = v_1$, $C(e_4) = v_2$, $C(e_5) = v_3$, $C(e_6) = v_4$, and $C(e_7) = v_5$ (using branch names as in Figure 5).
But a collector does not necessarily need to be connected to a node of the translinear digraph. It can also serve as a (current-mode) output of the network, or it can be connected directly to a voltage supply. We will express this by $C(e) = v_{\mathrm{ext}}$, imagining v_{ext} as an additional node outside of G.

Example 5: In Figure 4/Figure 5, $C(e_2) = C(e_3) = v_{\mathrm{ext}}$.
In summary: A translinear network topology is specified by

- a translinear digraph G
- and a map $C : E(G) \to V(G) \dot\cup \{v_{\mathrm{ext}}\}$, where C has the property that for each $e \in E(G)$, either $C(e) = v_{\mathrm{ext}}$ or

$$\mathrm{rank}(C(e)) > \mathrm{rank}(\mathrm{head}(e)). \tag{7}$$

We identify

a branch e	with	a transistor,
tail (e)	with	the transistor's base node,
head (e)	with	the transistor's emitter node,
and C (e)	with	the transistor's collector node.

We call the map C the **collector assignment** of the network. (The notion of a collector assignment will be refined in Section III-C.)

Remark 2: Note that in the latter example, $C(e_j) = \mathrm{tail}(e_j)$ for $j = 1, 4, 5, 6$, meaning that the collector is connected to the same node as the base of the respective transistor.
In such cases of diode-like transistor usage, the condition expressed by eqn. (7) is "automatically" satisfied, since by the definition of the rank of a node,

$$\mathrm{rank}(\mathrm{tail}(e)) = \mathrm{rank}(\mathrm{head}(e)) + 1.$$

The information given by a translinear digraph G and a collector assignment C is already a fairly complete description of a translinear network. In particular, a netlist for simulation or symbolic analysis of the network can be set up, if just some necessary "interface" information is added, for instance the connection of independent current sources which represent inputs of the network. These interfacing issues will be considered in Section III-C.

C. The Interface of a Translinear Network

This section addresses the question of what should be considered as the "interface" of a translinear network, i. e. of how inputs are applied to and outputs are supplied by the network.

Of course, inputs and outputs of a translinear circuit are in current mode.

1) Outputs: All collectors which are designated "external" by the collector assignment C (i. e., the collectors of those transistors for which $C(e) = v_{\mathrm{ext}}$) can be used as outputs of the network. But not only the current of a single collector, also sums or differences of them can be considered as outputs. In general, we consider a single symbolic output y of a network which is a sum of positive and negative collector currents:

$$y = \sum_{C(e)=v_{\mathrm{ext}}} \sigma(e) x_e, \tag{8}$$

where x_e denotes the collector current of the transistor corresponding to branch e and $\sigma(e) \in \{-1, 0, 1\}$.

Example 6: For the frequency doubling network (Figure 4), $\sigma(e_2) = 1$ and $\sigma(e_3) = -1$.
To avoid the usage of σ, we will henceforth use three symbolic nodes $v_{\mathrm{out+}}, v_{\mathrm{out-}}, v_{\mathrm{void}}$ instead of v_{ext} and work with the following refined concept of a collector assignment:

Definition 3: Let G be a translinear digraph. A **(refined) collector assignment on** G is a map

$$C : E(G) \to V(G) \,\dot\cup\, \{v_{\text{out+}}, v_{\text{out-}}, v_{\text{void}}\}$$

such that for every $e \in E(G)$:

$C(e) \in \{v_{\text{out+}}, v_{\text{out-}}, v_{\text{void}}\}$ or $\text{rank}(C(e)) > \text{rank}(\text{head}(e))$.
Thus, the symbolic output of a network will be (compare to (8)):

$$y = \sum_{C(e)=v_{\text{out+}}} x_e - \sum_{C(e)=v_{\text{out-}}} x_e. \tag{9}$$

2) Inputs and Ground Node Selection: In principle, an independent input current i_v can be applied to any node v of the translinear digraph G. The only restriction is that the sum of all input and output currents (the latter are now all collector currents with $C(e) \in \{v_{\text{out+}}, v_{\text{out-}}, v_{\text{void}}\}$) must always be zero, which cannot be satisfied if independent currents are prescribed for all nodes of G. Therefore, we select one particular node v_0 of G that serves as a "valve".

For convenience, we will always choose this "valve" node as the "reference" or "ground" node of the circuit. So we can simply assume that we have an independent current source connected to each node of G except one, which we call the ground node, and denote it by v_0. We call all other nodes "input nodes".

In the examples we have seen so far, most of the nodes of the translinear digraph have no current source connected to them. This amounts to an indepentent input which happens to be a constant zero and is not to be confused with the role of the ground node!

Example 7: For the frequency doubling network (Figure 4): $v_0 = v_6$; $i_{v_1} = u_1$, $i_{v_2} = u_2$, $i_{v_3} = i_{v_4} = i_{v_5} = 0$.
A triple (G, C, v_0) of a translinear digraph G, a collecor assignment C (in the refined sense) and a ground node v_0 is a network description which is complete in the sense that the network equations can entirely be set up. The resulting system of polynomial equations will be studied in Section V. That system contains all node equations except the one of v_0. For this reason, the current into the collector of the transistor corresponding to a branch e with $C(e) = v_0$ does not occur in the network equations. The same is true if $C(e) = v_{\text{void}}$, while in all other cases, the collector current belonging to e does affect the system: either in the node equation of $C(e) \in V(G) \setminus \{v_0\}$ or, if $C(e) \in \{v_{\text{out+}}, v_{\text{out-}}\}$, in the output equation.

Hence, for the system of network equations, it does not matter whether $C(e) = v_{\text{void}}$ or $C(e) = v_0$ for a branch e. To avoid redundant entries in our catalog, we do not allow $C(e) = v_0$ for any e. The simplest possibility to guarantee $C(e) \neq v_0$ for all e is to choose $v_0 \in V(G) \setminus \text{image}(C)$.

Thus we arrive at the following precise formal definition of a translinear network:

Definition 4: A **(static) formal translinear network** is a triple $N = (G, C, v_0)$ of a translinear digraph G, a (refined) collector assignment $C : E(G) \to V(G) \,\dot\cup\, \{v_{\text{out+}}, v_{\text{out-}}, v_{\text{void}}\}$ and a node $v_0 \in V(G) \setminus \text{image}(C)$. We call v_0 the **ground node of** N. The **number of transistors of** N is the number of branches of G.

For the rest of this paper, we denote the input nodes (see Subsection III-C.2) of a formal network $N = (G, C, v_0)$ by $v_1, \ldots, v_{n-1}$ (thus $V(G) = \{v_0, \ldots, v_{n-1}\}$) and the branches of G by $e_1, \ldots, e_b$.

IV. A Catalog of Translinear Network Topologies

For various reasons it may happen that the circuit modelled by a formal network (G, C, v_0) is of no practical value. For example, consider a *source* v of G, that is, a node which is not the head of any branch. No emitter is connected to v. Since we assume an independent input current applied to v unless $v = v_0$, we get into trouble if no collector is connected to v as well, because the bases cannot compensate the externally forced current. Consequently, we discard formal networks possessing a source v of G with $C(e) \neq v$ for all branches e.

Obviously, a network without any output is of no use. Therefore, we only consider collector assignments with $C(e) \in \{v_{\text{out+}}, v_{\text{out-}}\}$ for at least one branch e. Furthermore, if there is no branch e with $C(e) = v_{\text{out+}}$, the output of the network is

$$y = -\sum_{C(e)=v_{\text{out-}}} x_e$$

(compare eqn. (9)). We assume that for applications an inverted output $-y$ is equally of benefit as y is, so we can swap $v_{\text{out+}}$ and $v_{\text{out-}}$. Thus we can require $C(e) = v_{\text{out+}}$ for at least one branch e without losing any non-redundant formal network.

Both of the stated "practical value" conditions are incorporated into the following definitions.

Definition 5: We call a formal translinear network (G, C, v_0) **valid**, if $C(e) = v_{\text{out+}}$ for at least one $e \in E(G)$, and for each source v of G, either $v = v_0$ or there is at least one $e \in E(G)$ with $C(e) = v$.
A collector assignment is valid, if we can be sure that it leads to a valid formal network:

Definition 6: We call a collector assignment C on a translinear digraph G **valid**, if $C(e) = v_{\text{out+}}$ for at least one $e \in E(G)$ and there is at most one source v of G such that $v \notin \text{img}(C)$.
(In case there is exactly one such source node, it can and must be chosen as ground node.)

We have now provided a way to regard translinear networks as formal combinatorial objects. However, some of these have to be considered equivalent (or isomorphic), because they represent the same actual network. So only one representative of each equivalence class should be included in the catalog.

Definition 7: Two formal translinear networks (G, C, v_0) and (G', C', v_0') are called **isomorphic** if there exist a digraph automorphism $\varphi : G \to G'$ and a permutation σ on $V(G)$ such that σ permutes only the top layer of G and

$$\varphi(C'(e)) = C(\varphi(e))$$

whenever $\text{rank}(C'(e)) < R$ and

$$\sigma(C'(e)) = C(\varphi(e))$$

whenever $\text{rank}(C'(e)) = R$.

number of branches:	4	5	6	7	8	9
translinear digraphs:	2	3	19	39	174	559

TABLE I

NUMBER OF ISOMORPHISM CLASSES OF TRANSLINEAR DIGRAPHS FOR A GIVEN NUMBER OF BRANCHES.

number of transistors	pairs (G, C)	valid pairs
4	250	177
5	2248	1868
6	51930	42102
7	1149476	978812
8	32125843	27335135

TABLE II

COMBINATORIAL RESULTS CONCERNING A CATALOG OF TRANSLINEAR NETWORK TOPOLOGIES.

Thus we have specified what exactly we want to be included in the catalog: One representative from each isomorphism class. We devide the task of cataloging the formal translinear networks with a given number b of transistors into 3 sub-tasks:

1) Catalog the translinear digraphs with b transistors.
2) For a given translinear digraph G, catalog the collector assignments on G. (This is the most intricate of the 3 subtasks.)
3) For a given translinear digraph G and a collector assignment C, generate the networks (G, C, v_0). (This is trivial.)

For tasks 1 and 2, combinatorial methods based on *orderly generation* [18] have been adapted and implemented in C++. (For details see [11].) The software has been used to generate exhaustive and non-redundant lists of pairs (G, C). Tables I and II and Figure 7 show some statistics on the resulting catalog. The iteration for v_0 is performed "online" when a network is searched. Some timings appear at the end of VII.

Figure II in particular shows the numbers of pairs (G, C) in the catalog, and the numbers of those pairs where C is valid. For up to 6 transistors, Appendix shows how the numbers in the right column split up for the particular digraphs.

V. TRANSLINEAR NETWORK EQUATIONS

In this section, we investigate how the system of network equations looks like for a formal translinear network.

We remind that in general, the behavior of an electrical network is described by

- the loop equations (given by Kirchhoff's Voltage Law),
- the node equations (given by Kirchhoff's Current Law),
- and the element equations.

In the case of a translinear network, the loop equations and the element equations (namely the exponential transistor model) are merged by the translinear principle and result in the translinear loop equations.

Thus, the behavior of a static translinear network is entirely described by the translinear loop equations and the node equations.

A. Translinear Loop Equations

Let $N = (G, C, v_0)$ be a formal translinear network, and let x_j denote the collector current of the transistor corresponding to branch e_j, $j = 1, \ldots, b$. Consider a loop S of G. The translinear loop equation for S is

$$0 = B_S := \prod_{u_j > 0} x_j^{u_j} - \prod_{u_j < 0} x_j^{-u_j},$$

where

$$u_j := \begin{cases} 1, & \text{if } e_j \text{ is a forward branch of } S, \\ -1, & \text{if } e_j \text{ is a backward branch of } S, \\ 0, & \text{if } e_j \text{ is not a branch of } S. \end{cases}$$

The binomials B_S generate a very interesing algebraic structure called the **toric ideal** of G [22], denoted by I_G[4]. In [11], a procedure to compute I_G has been derived.

B. Node Equations

Consider an input node $v_j \in V(G)$, $1 \le j \le n-1$, of a formal translinear network $N = (G, C, v_0)$. Since we neglect the base currents of the transistors, the only contributions to the node equation of v_j come from emitters, collectors, and the independent input current i_j (in Subsection III-C.2 denoted by i_{v_j}). Thus, the node equation of v_j is

$$0 = i_j + \sum_{\substack{1 \le k \le b \\ \text{head}(x_k) = v_j}} x_k - \sum_{\substack{1 \le k \le b \\ C(x_k) = v_j}} x_k.$$

C. The Network Ideal

For a formal translinear network $N = (G, C, v_0)$, we consider the system consisting of the translinear loop equations, the node equations, and the output equation (compare eqn. (9))

$$0 = y - \sum_{C(e_k) = v_{\text{out+}}} x_k + \sum_{C(e_k) = v_{\text{out-}}} x_k.$$

We call the ideal generated by the corresponding polynomials the *network ideal*:

Definition 8: Let $N = (G, C, v_0)$ be a formal static translinear network. The **network ideal**

$$I_N \subset \mathbb{Q}[x_1, \ldots, x_b, i_1, \ldots, i_{n-1}, y]$$

is the ideal generated by

- the (non–linear) toric ideal
$$I_G \subset \mathbb{Q}[x_1, \ldots, x_b] \subset \mathbb{Q}[x_1, \ldots, x_b, i_1, \ldots, i_{n-1}, y]$$
- the (linear) polynomials $j = 1, \ldots, n-1$
$$-i_j + \sum_{\text{head}(e_k) = v_j} x_k - \sum_{C(e_k) = v_j} x_k \qquad (10)$$
- and the (linear) polynomial
$$-y + \sum_{C(e_k) = v_{\text{out+}}} x_{e_k} - \sum_{C(e_k) = v_{\text{out-}}} x_{e_k}.$$

[4]Recall that an ideal I in the polynomial ring $\mathbb{Q}[x_1, \ldots, x_n]$ generated by $f_1, \ldots, f_k$ consists of all linear combinatorics $g_1 f_1 + \cdots g_k f_k$ with $g_i \in \mathbb{Q}[x_1, \ldots, x_n]$.

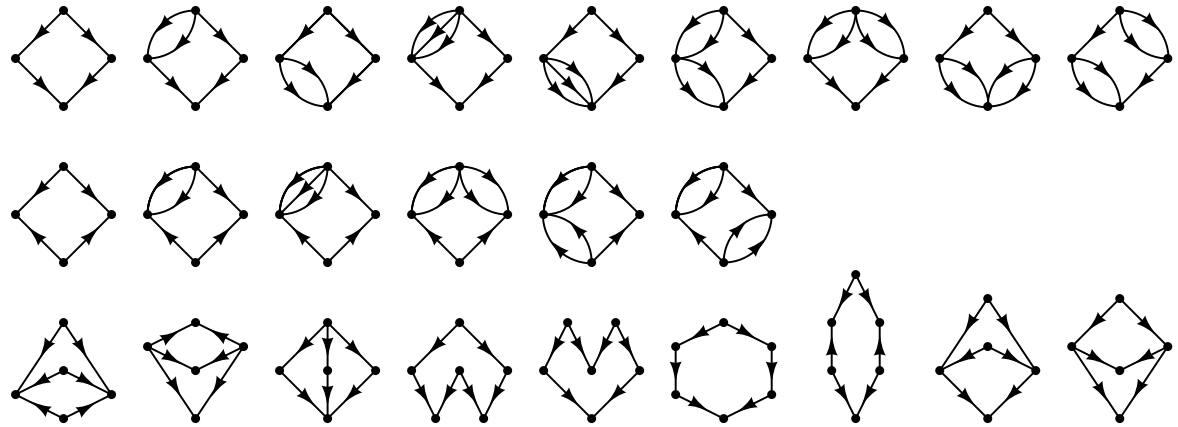


Fig. 7. The translinear digraphs with 6 or less branches.

The following motivation to consider the ideal I_N might be in order. Let us denote, for the moment, the above given generators of I_N by $f_1, \ldots, f_k \in \mathbb{Q}[x_1, \ldots, x_b, i_1, \ldots, i_{n-1}, y]$. Then the variables $\mathbf{x} = (x_1, \ldots, x_b), \mathbf{i} = (i_1, \ldots, i_{n-1}), y$ have to satisfy the equations $f_1(\mathbf{x}, \mathbf{i}, y) = \cdots = f_k(\mathbf{x}, \mathbf{i}, y) = 0$ and, as a consequence, the equation $f(\mathbf{x}, \mathbf{i}, y) = 0$ for arbitrary $f = g_1 f_1 + \cdots + g_n f_n$ in I_N. Thus, I_N codifies algebraically all information about the solutions of $f_1 = \cdots = f_k = 0$. To consider the ideal I_N has several advantages. First, we may replace $f_1, \ldots, f_k$ by other generators of I_N which might be easier to solve without changing the solution set but . For linear systems the Gauß'ian algorithm does exactly this, for non–linear polynomial systems we use Gröbner bases algorithms, cf. [9].

Another advantage is that we can find in I_N "hidden constraints", that is relations between the input currents $i_1, \ldots, i_{n-1}$ and the output y. These are exactly the polynomials in I_N not depending on $x_1, \ldots, x_b$, denoted by $I_{N'}$ below. The computation of these elements can again be done with Gröbner bases and the process is called elimination (of $x_1, \ldots, x_b$), cf. [9].

D. Elimination of Collector Currents

As inputs of a formal translinear network we have the input currents $i_1, \ldots, i_{n-1}$, the output is y. To get a direct input-output relationship, we eliminate the remaining variables, the collector currents $x_1, \ldots, x_b$, from the network ideal. We denote the resulting ideal by I'_N:

$$I'_N := I_N \cap \mathbb{Q}[i_1, \ldots, i_{n-1}, y].$$

The computer algebra system SINGULAR [8] has been used to compute I_N and to perform the elimination for each formal network in the catalog with 7 or less transistors. It has been found that whenever the inputs $i_1, \ldots, i_{n-1}$, expressed as linear combinations of the collector currents via eqn. (10), are linearly independent, I'_N turns out to be a principle ideal. In other words, in these cases the elimination of collector currents always yielded a single polynomial equation $f_N(i_1, \ldots, i_{n-1}, y) = 0$ in the input and output currents.

The catalog of formal networks has been equipped with a generator $f_N \in \mathbb{Z}[i_1, \ldots, i_{n-1}, y]$ of I'_N for every formal network N. Since I_N is homogeneous and prime, so are I'_N and f_N.

Remark 3: Depending on which collectors are assigned to $v_{\text{out+}}$ and $v_{\text{out-}}$, it may happen that the translinear loop equations are simply thrown away during the elimination. This is the case if and only if f_N is linear and represents a kind of degeneration of the network N: The input-output relation is not at all influenced by the translinear loops! We can ignore these degenerated networks, the linear behavior can as well be achieved by pure current addition/substraction via node equations, without any transistors.

VI. THE CATALOG AS A SYNTHESIS TOOL

A. The Input Matrix

Assume a circuit is to be designed whose desired behavior is given by a polynomial equation $g(u_1, \ldots, u_s, y) = 0$, where $u_1 \ldots, u_s$ are input variables and y is an output variable. (For simplicity we restrict to the case of only one output.) We assume that g has integer coefficients and is irreducible and homogeneous.

We want to test all formal translinear networks from our catalog whether they can be used to implement g. Since we have computed for each formal network $N = (G, C, v_0)$ in the catalog a polynomial $f_N \in \mathbb{Z}[i_1, \ldots, i_{n-1}, y]$ describing its behavior, we test the suitability of a particular formal network N by testing whether f_N "matches" g.

This matching test is not as simple as it appears: We must clarify how to identify the inputs $u_1, \ldots, u_s$ with the inputs currents $i_1, \ldots, i_{n-1}$ of the formal network.

We assume that for each input u_k, we can use as many current sources as we want, each delivering the same current u_k. In our model, each of these current sources is connected between the ground node v_0 and one of the input nodes $v_1, \ldots, v_{n-1}$. Both orientations are possible, so that the current source contributes either u_k or $-u_k$ to the respective input current i_j. Of course, several sources can be connected to one input node, even several sources with the same input current

u_k. Thus, the relation of the available inputs $u_1, \ldots, u_s$ with the formal node inputs $i_1, \ldots, i_{n-1}$ is

$$i_j = \sum_{k=1}^{s} b_{jk} u_k$$

for $j = 1, \ldots, n-1$ with $b_{jk} \in \mathbb{Z}$. Without additional circuitry, an input node cannot be supplied with anything else than a finite sum of the currents delivered by the available sources.

We call the matrix $B := (b_{jk}) \in \mathbb{Z}^{(n-1)\times s}$ the **input matrix.**

To make the polynomials f_N and g "comparable", we map f_N to $\mathbb{Z}[u_1, \ldots, u_s, y]$ via

$$\begin{aligned} \varphi_B : \mathbb{Z}[i_1, \ldots, i_{n-1}, y] &\to \mathbb{Z}[u_1, \ldots, u_s, y], \\ i_1 &\mapsto b_{1,1}u_1 + \cdots + b_{1,s}u_s, \\ &\vdots \\ i_{n-1} &\mapsto b_{n-1,1}u_1 + \cdots + b_{n-1,s}u_s, \\ y &\mapsto y. \end{aligned}$$

We can now formulate precisely what we mean by a match of f_N and g:

Definition 9: Let $f_N \in \mathbb{Z}[i_1, \ldots, i_{n-1}, y]$ and $g \in \mathbb{Z}[u_1, \ldots, u_s, y]$. A **match of f_N onto g** is a matrix $B \in \mathbb{Z}^{(n-1)\times s}$ such that $g = \lambda\varphi_B(f_N)$ for some $\lambda \in \mathbb{Q} \setminus \{0\}$. (Obviously, we allow multiplication by a nonzero constant λ because $g = 0 \Leftrightarrow \lambda g = 0$.)

For practical reasons, we restrict to matches where the coefficients b_{jk} are bounded by an integer l, usually $l = 1$ or $l = 2$.

B. Gröbner Bases and Matching Problem

By the *matching problem*, we mean the problem to determine all matches $B \in \{-l, -l+1, \ldots, l-1, l\}^{(n-1)\times s}$ of f_N onto g.

If the total degree of g does not coincide with the total degree of f_N, we know that no match of f_N onto g exists. From now on we assume equality of the degrees and define $\deg g = \deg f_N =: d$.

We solve the matching problem algorithmically by comparison of coefficients: $g = \lambda\varphi_B(f_N)$ if and only if every coefficient of the polynomial $g - \lambda\varphi_B(f_N)$ vanishes. Since the entries of B are unknown, we treat them as symbolic variables. Then, regarding the coefficients of $g - \lambda\varphi_B(f_N)$ as polynomials in the entries of B and in λ, we get a system of polynomial equations which must be solved. The integer solutions are the entries of those matrices B which are a match.

Formally, we introduce indeterminates Λ for λ and $\mathbf{w} := \{w_{jk}\}_{\substack{j=1,\ldots,n-1 \\ k=1,\ldots,s}}$ for the entries $b_{jk}$of B and consider the homomorphism of rings

$$\begin{aligned} \varphi : \mathbb{Q}[i_1, \ldots, i_{n-1}, y] &\to \mathbb{Q}[u_1, \ldots, u_s, y, \mathbf{w}, \Lambda] \\ i_j &\mapsto \sum_{k=1}^{s} w_{jk}u_k. \end{aligned}$$

The ideal

$$I_{\text{coef}} := \langle \text{coef}(\Lambda\varphi(f_N) - g, M) \,|\, M \in \text{Mon}_d(u_1, \ldots, u_s, y) \rangle$$

in $\mathbb{Q}[\mathbf{w}, \Lambda]$ (where $\text{Mon}_d(u_1, \ldots, u_s, y)$ is the set of monomials of degree d in $u_1, \ldots, u_s, y$) represents the system of equations which the entries of B have to satisfy to be a "match".

To force the entries of B to be integers in the range $-l, \ldots, l$, we demand additionally

$$w_{jk} \cdot (w_{jk} - 1)(w_{jk} + 1) \cdot \cdots \cdot (w_{jk} - l)(w_{jk} + l) = 0$$

for $j = 1, \ldots, n-1$ and $k = 1, \ldots s$.

Hence, we need to compute the zero set $Z(I_0)$ of the ideal

$$I_0 := \left\langle I_{\text{coef}}, \left\{ \prod_{\nu=-l}^{l} (w_{jk} - \nu) \right\}_{j,k} \right\rangle \subset \mathbb{Q}[\mathbf{w}, \Lambda].$$

Since $\left\{\prod_{\nu=-l}^{l}(w_{jk} - \nu)\right\}_{j,k} \subset I_0$, the ideal I_0 is zero-dimensional and there are no irrational solutions.

We can compute these points of $Z(I_0)$ using Gröbner basis methods. For example, a computation of the minimal associated primes of I_0 reveals maximal ideals representing the points of $Z(I_0)$ and thus the matches we are looking for.

To increase efficiency, the computation can be performed over a finite field $\mathbb{Z}/p$ instead of $\mathbb{Q}$. In this case, it is possible that solutions over Z/p appear which are no solutions over $\mathbb{Q}$. This can be avoided by choosing the prime number p sufficiently large.

A SINGULAR procedure has been written that determines all matches of a given network polynomial f_N onto a given "target behavior" polynomial g, see [11].

It was somewhat surprising that the sophisticated algorithm to compute minimal associated primes (`minAssGTZ` in SINGULAR) from abstract algebraic geometry together with modular computation was an effective way to solve the matching problem, even in industrial applications.

C. Final Network Check

Assume we have found a match, so that we know a formal network $N = (G, C, v_0)$ and an input matrix B such that $\lambda\varphi_B(f_N) = g$, that is, the pair (N, B) shows exacly the behavior we want.

In the system of network equations (see Section V), we can replace i_j by $\sum_{k=1}^{s} b_{jk}u_k$ for $j = 1, \ldots, n-1$ and solve it for the collector currents $x_1, \ldots, x_b$, obtaining them as functions of $u_1, \ldots, u_s$. The result is, for each collector current x_ν, a finite number of solutions

$$x_\nu^{(1)}(u_1, \ldots, u_s), \ldots, x_\nu^{(d_\nu)}(u_1, \ldots, u_s).$$

To find out which one of these solutions corresponds to the actual current of the respective transistor, we perform the following check.

1) Positivity Check of Collector Currents : All collector currents must be positive, otherwise our transistor model (eqn. (1)) is invalid.

Usually, the specification of the desired network behavior includes a range $u_k^{\min} < u_k < u_k^{\max}$ for each input, $u_k^{\min} \in \mathbb{R} \cup \{-\infty\}$, $u_k^{\max} \in \mathbb{R} \cup \{\infty\}$. Given these ranges, we can check for each collector current x_ν which of the solutions $x_\nu^{(\xi)}$ violate the condition $x_\nu^{(\xi)}(u_1, \ldots, u_s) > 0$.

In all example computations it was found that there is at most one $\xi \in \{1,\dots,d_\nu\}$ with $x_\nu^{(\xi)}(\mathbf{u}) > 0$ for all $\mathbf{u} \in]u_1^{\min}, u_1^{\max}[\times\cdots\times]u_s^{\min}, u_s^{\max}[$. If there is none, a hardware implementation of the network will not work, because the model assumption of positive collector currents is not satisfied. We can delete the pair (N, B) from our set of "candidates".

But if there is indeed a positive solution for each collector current, we have determined the exact explicit dependence of the collector currents on $u_1,\dots,u_s$, and thus also the exact explicit dependence of the output

$$y = \sum_{C(e)=v_{\text{out+}}} x_e - \sum_{C(e)=v_{\text{out-}}} x_e$$

on $u_1,\dots,u_s$.

2) Output Function Check: After the positivity check of collector currents, we have an explicit description of the network behavior in the form $y = h(u_1,\dots,u_s)$, and we know that $g(u_1,\dots,u_s,h(u_1,\dots,u_s)) = 0$. Still, it might be that the intended behavior was a different branch of the solution of the polynomial equation $g = 0$. (For example, we may look for a network with $y = \sqrt{u_1}$ and thus specify $g = y^2 - u_1$. Then the network search may yield a network with $y = -\sqrt{u_1}$. Of course, less pathological examples exist where the difference is more than the sign.)

The comparison of h with the intended explicit behavior is our last check to filter out formal networks which are of no practical relevance.

Both the positivity check of collector currents and the output function check have been implemented in MATHEMATICA.

D. Shift of the Output by Inputs

In applications, the notion of a "match" used before is too restrictive:
When a network for an output function $y = h(u_1,\dots,u_s)$ is searched, it is reasonable to look for networks implementing not only $y = h(u_1,\dots,u_s)$, but also $y = h(u_1,\dots,u_s) - \sum_{j=1}^s d_j u_j$ with $d_1,\dots,d_s \in \mathbb{Z}$, and shift the output of such a network by $\sum_{j=1}^s d_j u_j$ to obtain the desired output current. (Current-mode summation being one of the most simple operations to implement.)

Therefore we consider "d-matches":

Definition 10: Let $f \in \mathbb{Z}[i_1,\dots,i_{n-1},y]$, $g \in \mathbb{Z}[u_1,\dots,u_s,y]$ and $d \in \mathbb{Z}^s$. A **d-match of f onto g** is a matrix $B \in \mathbb{Z}^{(n-1)\times s}$ such that $g = \lambda\varphi_{B,u}(f_N)$ for some $\lambda \in \mathbb{Q}^*$, where $\varphi_{B,u}$ is defined by

$$\begin{aligned}
\varphi_{B,u}(i_1) &= b_{1,1}u_1 + \cdots + b_{1,s}u_s,\\
&\vdots\\
\varphi_{B,u}(i_{n-1}) &= b_{n-1,1}u_1 + \cdots + b_{n-1,s}u_s,\\
\varphi_{B,u}(y) &= y - d_1u_1 - \cdots - d_su_s.
\end{aligned}$$

A "match" as defined before coincides with a 0-match.

VII. EXAMPLE APPLICATION

The following example comes from industrial applications at the company Analog Microelectronics GmbH (in the following: AMG). The AMG specializes in the development of industrial and automotive electronics and sensor systems. The semiconductor processes used in these areas are naturally different from areas with pure digital processes. Whereas with pure digital processes power dissipation and speed are the crucial criteria, in the area of automotive and industrial electronics further criteria are relevant: higher voltages, Signal/Noise Ratio, ESD protection, matching, or even integrated MEMS. The used processes are usually modular and it is possible to stack 3 or 4 VBE, such that translinear networks can be an ideal framework for modelling processes in these fields.

In the following example we adress the problem whether the nonlinear behavior of a certain sensor device can be compensated by a static translinear circuit (to be used for the construction of a display unit). The problems concern the question whether the nonlinear behavior of a certain sensor device can be compensated by a static translinear circuit.

A humidity sensor subsystem for air conditioning is in development at AMG. The sensor device to be used is optimized for an operating temperature of 23°C; at this temperature, its output is virtually proportional to the relative humidity. For other temperatures, it shows deviations from this linear behavior which have to be compensated. For this purpose, an "analog computation" circuit is desired to reconstruct the deviation. The circuit of course needs a second input where information about the actual temperature is provided independently of the sensor output.

H	-10°C	23°C	50°C	70°C	100°C	140°C
10%	-1%	0%	1%	2%	3%	4%
20%	-1.5%	0%	1.5%	3%	4.48%	6%
30%	-1.63%	0%	1.82%	3.55%	5.55%	7.43%
40%	-1.74%	0%	2.08%	4%	6.31%	8.35%
50%	-1.78%	0%	2.2%	4.2%	6.78%	8.78%
60%	-1.78%	0%	2.27%	4.35%	6.94%	9%
70%	-1.78%	0%	2.31%	4.43%	7%	9%
80%	-1.78%	0%	2.31%	4.51%	7.12%	9%
90%	-1.78%	0%	2.31%	4.51%	7.12%	9%

TABLE III
MEASURED OUTPUT DEVIATION OF HUMIDITY SENSOR.

We denote the real relative humidity by H, the output of the sensor (the "measured humidity") by H_m. Table III shows the output deviation $H_m - H$ at some temperatures.

Since the desired circuit shall "compute" $H_m - H$ from H_m (and the temperature T), the first task was to find an algebraic function f_{spec} as specification of a "translinear synthesis" problem, so that these points satisfy (or are close to)

$$H_m - H = f_{\text{spec}}(H_m, T).$$

"Educated guessing" and heavy usage of the MATHEMATICA-Functions `FindMinimum` and `ProjectiveRationalize` showed that $f_{\text{spec}}(H_m,T) = \lambda(T)y(H_m)$ with

$$\lambda(T) = -0.0012167\,(t-10)(t-1)(t+3), \qquad t = \frac{T}{23^\circ\text{C}},$$

$$y(H_m) = \frac{hH_m - H_m^2 + H_m\sqrt{5h^2 - 2hH_m + H_m^2}}{2h}, \quad (11)$$

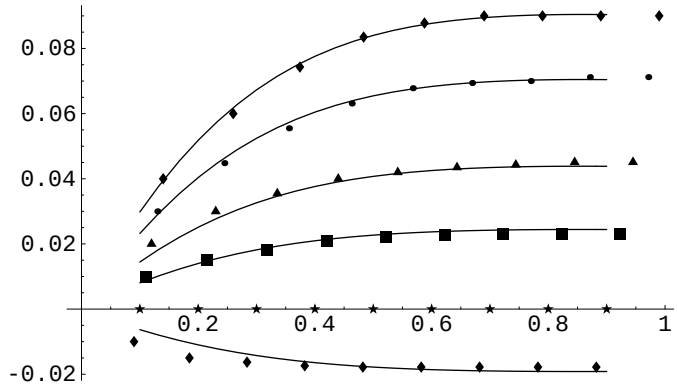


Fig. 8. A plot of $\lambda(T)y(H_m)$ and the points $(H_m, H_m - H)$ for $T = -10°\mathrm{C},\ 0°\mathrm{C},\ 23°\mathrm{C},\ 50°\mathrm{C},\ 70°\mathrm{C},\ 100°\mathrm{C}$.

and $h = 0.3294$ approximates the points very nicely. The function y is one of the two solutions of the implicit polynomial equation

$$hH_m^2 + hH_m y - H_m^2 y - hy^2 = 0. \tag{12}$$

Figure 8 shows the high quality of the approximation.

λ consists of simple multiplications and can be implemented easily by translinear as well as other analog circuitry, we don't go into further details of this part of the design problem. Instead we concentrate on the synthesis of a subcircuit implementing y, where the tools of this paper have been applied succesfully.

The implicit description of the desired network behavior is given by (compare eqn. (12))

$$0 = g := u_1^2 u_2 + u_1 u_2 y - u_1^2 y - u_2 y^2.$$

The inputs are $u_1 = H_m$ and $u_2 = h$. (We use a stationary input current source for u_2 to supply the constant h to the network.) We wish to implement $g(u_1, u_2, y) = 0$ by a circuit with at most 6 transistors.

15 of the 24 translinear digraphs with 6 or less branches (see Fig. 7) consist of only one loop of length 4 and some parallel branches. For these 15 digraphs, the toric ideal has exactly one generator of degree 2 and no generator of higher degree. Consequently, for all formal networks $N = (G, C, v_0)$ where G is one of these digraphs, $\deg f_N = 2 < 3 = \deg g$, so the networks cannot be used for implementing g.
There are 9 digraphs remaining with 6 branches each. (In Appendix , they carry the indices 6, 7, 8, 9, 10, 19, 20, 21, and 24.) A search for matches has been conducted through all formal networks based on any of these 9 digraphs. As bound for the entries of the input matrix, $l = 2$ was chosen.

For each of the 9 digraphs, Table IV shows the number of solution triples (N, d, B) of a formal translinear network N, an output shift vector d and an input matrix B which have been found. The second column shows the number of formal matches where $\varphi_{B,u}(f_N) = g$ (see Section VI-A), the third column shows how many of these triples passed the test for positive collector currents and correct explicit output function (see Subsection VI-C). The fourth column shows the approximate duration of the match search through all formal networks of the corresponding digraph (on a 1 GHz Pentium III processor) . The duration of the final network check, in comparison, can be neglected. (For digraph 6, it took 0.2 seconds; for digraph 24, it took approximately 10 minutes.)

All generated netlists implement the prescribed polynomial and allows the developer to select those which are optimal from aspects of expense and realistic behavior. All non–idealities for a concrete circuit is treated on the basis of this netlist. For example, global production tolerances like current gain or sheet resistances and local disturbance like mismatch or thermal gradients may affect the properties of the transfer function. This analysis is standard and has to be done using a spice–type simulator and the appropriate libraries containing information about the global and local process tolerances.

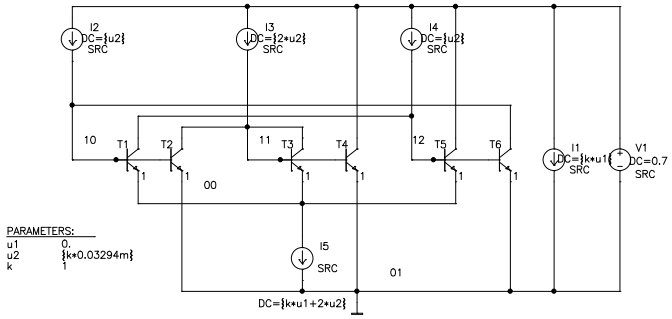


Fig. 9. A network based on digraph 6.

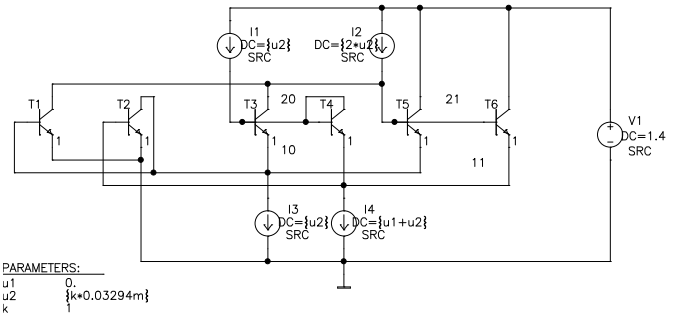


Fig. 10. A network based on digraph 9.

digraph index	solutions (N, d, B)	"relevant" solutions	search duration
6	3	1	5 sec
7	9	0	6.5 min
8	2	0	8 sec
9	94	10	5.3 min
10	333	49	6.5 hours
19	40	9	8 sec
20	870	167	2 days
21	387	0	5 sec
24	1542	203	6 days

TABLE IV

NUMBERS OF TRANSLINEAR NETWORKS FOUND FOR $g = u_1^2 u_2 + u_1 u_2 y - u_1^2 y - u_2 y^2$

.

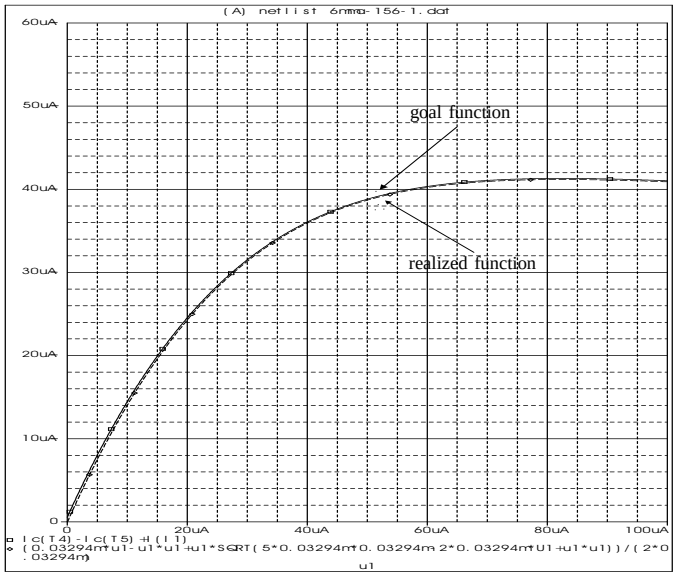


Fig. 11. Simulation result of the network of Figure 9.

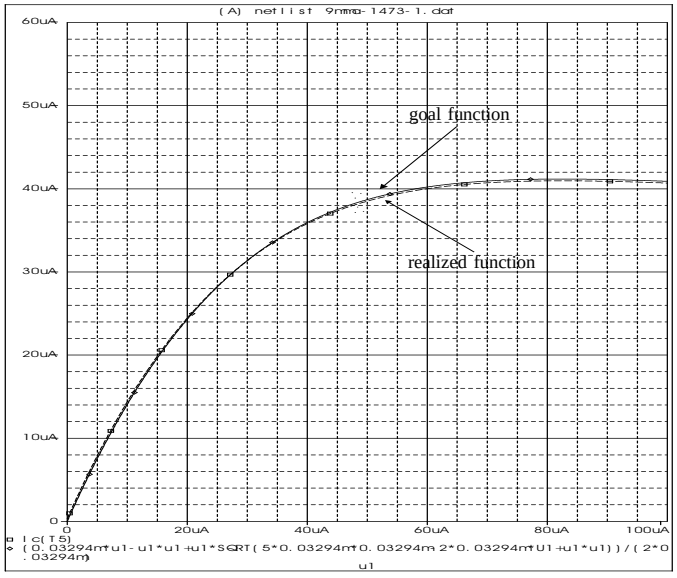


Fig. 12. Simulation result of the network of Figure 10.

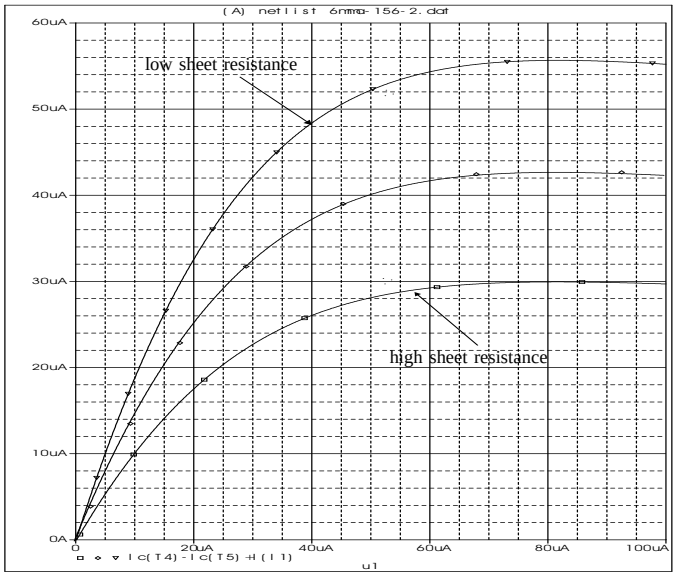


Fig. 13. Impact of variance of sheet resistance of the networkof Figure 9.

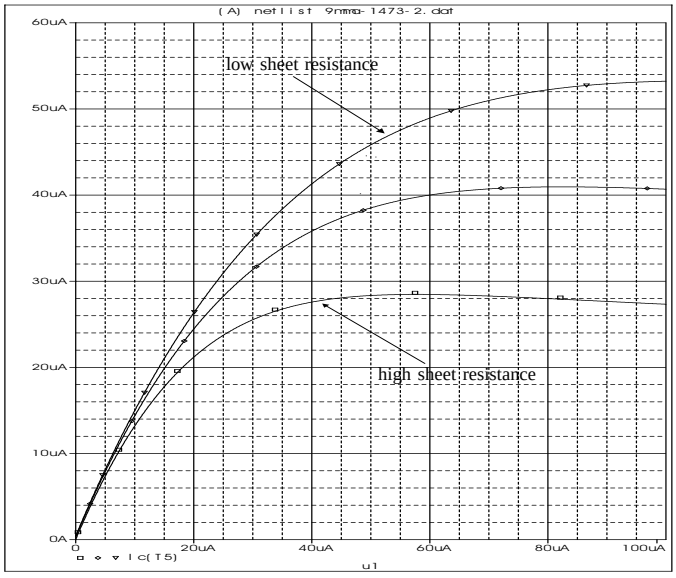


Fig. 14. Impact of variance of sheet resistance of the network of Figure 10.

A very instructive example is shown in Fig. 11-14. For two of the netlists, Fig. 11 and 12 show the results of a SPICE-simulation with Gummel-Poon transistor models, which are usual models during development for a specific semiconductor process. Both plots show the good compliance with the ideal behavior described by eqn. (11).

The currents used in the interface are typically derived for sheet resistance which varies within $\pm 30\%$ during production. In Fig. 13 and 14 the simulation for the two netlists show the impact of the sheet resistance on the output function. This impact is due to the actual transistor properties like e.g. early voltage and base current which take affect depending on the symmetry of the network.

In Fig. 13 the plotted curves differ just by a constant scaling factor, while in Fig. 14 the curves have significantly different gradients. This shows that the impact of sheet resistance is of much less significance for the network in Fig. 9 than for the network in Fig. 10. Therefore, the first network is the preferred candidate for hardware implementation.

Complexity and costs of the computations

Since space does not permit to detail the used combinatorial and algebraic algorithms we comment only on the complexity and the computational costs.

The graph algorithms to compute the pairs (C, G) in Table II, that is to compute the complete translinear network catalog, is exponential in the number of transistors. However, this has to be computed only once and then stored. This was done for up to 8 transistors which is sufficient for practical applications of this method (so far even 6 transistors were sufficient for all functions we tried). It took about 1 minute to generate all networks up to 6 transistors, 15 minutes for up to 7 transistors and 10 hours for up to 8 transistors (computations on a 2 GHz

Athlon XP 2800 processor).

The elimination from Section V.D and the matching problem from Section VI.B require Gröbner basis computations which are also exponential in the input. The concrete computation of the network relations f_N for all N in the catalog, requires about the same time as the computation of the catalog. But again, this has to be done only once.

The matching problem, however, has to be solved for every g (the time is more or less the same for all g with the same number of variables and degree). For a given g we have to check for all N in the catalog whether there exist λ and φ such that $g = \lambda\varphi(f_N)$. For fixed N this requires symbolic solving of a polynomial system with $\binom{n+d-1}{d} + ns + 1$ equations (where $\binom{n+d-1}{d}$ are of degree 2, and $ns+1$ of degree $2l+1$) in $ns+1$ variables ($n = \#$ nodes, $d = \deg(g), s+1 = \#$ variables on which g depends, $[-l,l] =$ range of integer solutions). In table IV we have $n = 5$ or 6, $d = 3$, $s = 2$, $l = 2$, that is we have to solve a system with 46 or 69 equations (of degree 2 and 5) in 11 or 13 variables as many times as there are formal networks (this number is shown in the fourth column of the table in the appendix; it is e.g. 3757 for digraph index 10 or 11602 for digraph index 24). Here the symbolic solving is nontrivial and the number of times this has to be done is big. This can actually not be computed with a standard general purpose system. We had to use the very fast system SINGULAR, the sophisticated algorithm `minAssGTZ` and modular computations to perform a single prime decomposition within a reasonable (several seconds) time. The number of times to do this is, although big, not a principal obstacle because it can be easily parallelized. The total (sequential) time is shown in table IV.

VIII. CONCLUSION

The research reported about in this paper provides

- a coherent mathematical "translinear network theory", consisting of a graph theoretic modelling framework for the topology of translinear circuits, and an analysis of the system of equations describing their behavior,
- the concept of a catalog of topologies for translinear networks as a resource for circuit design,
- algorithms to build such a catalog and to search it for a network complying with a particular behavior,
- and implementations of the algorithms, resulting in a software toolbox for translinear network synthesis.

As result, an exhaustive catalog of all static formal translinear networks with at most 8 transistors is available.

The details and implementations of the algorithms are worked out only for static networks, but can be adopted for dynamic networks as well.

While the implementation of the combinatorial algorithms is stand-alone software written "from scratch" in C++, the implementation of the algebraic algorithms, namely the symbolic treatment of the network equations and the match finding, heavily rely on the sophisticated Gröbner basis engine of SINGULAR and thus on more than a decade of experience contained in a special-purpose computer algebra system.

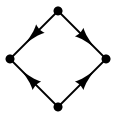
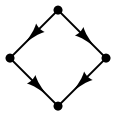
The application reported on in Section VII proves the practical applicability of the developed synthesis approach by using modern computer algebra with its highly developed systems and algorithms.

Altogether, the key role of translinear circuits as systematically designable nonlinear circuits is confirmed and strengthened.

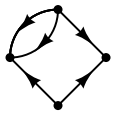
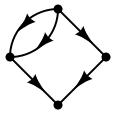
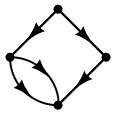
APPENDIX
THE STATIC FORMAL TRANSLINEAR NETWORKS WITH UP TO 6 TRANSISTORS

There are 2 translinear digraphs with 4 branches, 3 with 5 branches, and 19 with 6 branches, where only the first 11 are shown. For each of these digraphs, the following tables show in the third column the number of different (with respect to isomorphism, see Definition 7) valid collector assignments on G and in the fourth column the number of formal translinear networks (G, C, v_0) showing a nonlinear input-output relation (see Remark 3).

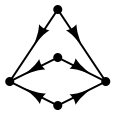
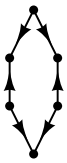
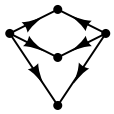
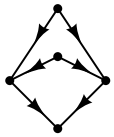
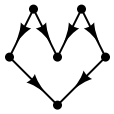
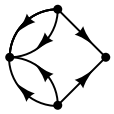
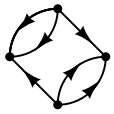
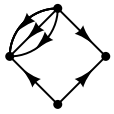
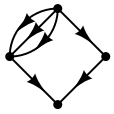
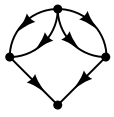
Networks with 4 transistors:

index	translinear digraph	collector assignments	formal networks
1		35	29
2		142	144

Networks with 5 transistors:

index	translinear digraph	collector assignments	formal networks
3		388	358
4		600	660
5		880	1083

Networks with 6 transistors:

index	translinear digraph	collector assignments	formal networks
6		300	291
7		537	398
8		413	628
9		1974	2593
10		4444	3757
11		661	577
12		661	727
13		661	727
14		959	910
15		1088	1235
16		754	867

REFERENCES

[1] Analog insydes. http://www.analog-insydes.de, 2001.

[2] Ying Dong, Sumit Bagga, and Wouter A. Serdijn. An inherently linear CMOS multiplier. In *Proc. ProRISC*, pages 483–486, Veldhoven, Nov 2004.

[3] R. Genin and R. Konn. Sinusoidal frequency doubler. *Electronic Letters*, 15(2):47–48, January 1979.

[4] Barrie Gilbert. Translinear circuits: A proposed classifi cation. *Electronics Letters*, 11(1):14–16, January 1975.

[5] Barrie Gilbert. Current-mode circuits from a translinear viewpoint: A tutorial. In C. Toumazou, F. J. Lidgey, and D. G. Haigh, editors, *Analogue IC design: The Current-mode Approach*, volume 2 of *IEE Circuits and Systems Series*, kapitel 2. Peter Peregrinus, London, 1990.

[6] Barrie Gilbert. Translinear circuits: An historical overview. *Analog Integrated Circuits and Signal Processing*, 9(2):95–118, March 1996.

[7] Barrie Gilbert. Translinear circuits. In John G. Webster, editor, *Wiley Encyclopedia of Electrical and Electronics Engineering*, volume 22 [Th-Un], pages 442–461. Wiley, New York, 1999.

[8] G.-M. Greuel, G. Pfi ster, and H. Schönemann. SINGULAR 2.0. A Computer Algebra System for Polynomial Computations, Centre for Computer Algebra, University of Kaiserslautern, 2001. `http://www.singular.uni-kl.de`.

[9] Gert-Martin Greuel and Gerhard Pfi ster. *A* SINGULAR *introduction to commutative algebra*. Springer, Berlin, 2002.

[10] David Ilsen. Cataloging translinear networks. In *Proc. 8th International Workshop on Symbolic Methods and Applications to Circuit Design (SMACD 2004)*, pages 32–35, Wroclaw, September 2004.

[11] David Ilsen. *Algebraic and Combinatorial Algorithms for Translinear Network Synthesis*. PhD thesis, Technische Universität Kaiserslautern, Kaiserslautern, Germany, 2006.

[12] David Ilsen and Ernst Josef Roebbers. Translinear networks from a combinatorial viewpoint. In *Proc. ProRISC*, pages 524–528, November 2004.

[13] Dieter Jungnickel. *Graphs, Networks and Algorithms*. Springer, Berlin, 1999.

[14] Eric A. M. Klumperink. *Transconductance Based CMOS Circuits*. PhD thesis, University of Twente, 1997.

[15] Eric A. M. Klumperink, Federico Bruccoleri, and Bram Nauta. Finding all elementary circuits exploiting transconductance. *IEEE Transactions on Circuits and Systems II*, 48(11):1039–1053, November 2001.

[16] Jan Mulder, Wouter A. Serdijn, Albert C. van der Woerd, and Arthur H. M. van Roermund. *Dynamic Translinear and Log-Domain Circuits – Analysis and Synthesis*. Kluwer Academic Publishers, Boston, 1999.

[17] Jan Mulder, Albert C. van der Woerd, Wouter A. Serdijn, and Arthur H. M. van Roermund. General current-mode analysis method for translinear fi lters. *IEEE Transactions on Circuits and Systems I*, 44(3):193–197, 1997.

[18] Ronald C. Read. *Every one a winner*, volume 2 of *Ann. Discrete Math.*, pages 107–120. North-Holland, Amsterdam, 1978.

[19] Jürgen Schmitz. Systematic generation of VCCS circuit topologies with cospectral and isomorphic multigraphs. In *Proc. 7th International Workshop on Symbolic Methods and Applications to Circuit Design (SMACD 2004)*, pages 25–30, Sinaia, October 2002.

[20] Jürgen Schmitz. On determining "useful" VCCS circuit topologies. In *Proc. 8th International Workshop on Symbolic Methods and Applications to Circuit Design (SMACD 2004)*, pages 52–55, Wroclaw, September 2004.

[21] Evert Seevinck. *Analysis and Synthesis of Translinear Circuits*, volume 31 of *Studies in Electrical and Electronic Engineering*. Elsevier, Amsterdam, 1988.

[22] Bernd Sturmfels. *Gröbner Bases and Convex Polytopes*, volume 8 of *University lecture series*. American Mathematical Soc., Providence, 1997.

[23] Rafael Vargas-Bernal, Arturo Sarmiento-Reyes, and Wouter A. Serdijn. Identifying translienar loops in the circuit topology. In *proc. IEEE International Symposium on Circuits and Systems II*, pages 585–588, 2000.

[24] Remco J. Wiegerink. *Analysis and Synthesis of MOS Translinear Circuits*. Kluwer, Boston/Dortrecht/London, 1993.

[25] Stephen Wolfram. *The mathematica book*. Wolfram Media, Champaign, Ill., 4. edition, 1999.