

Multiple DC Solution Determination using VHDL-AMS

Joachim Haase

Fraunhofer-Institut Integrierte Schaltungen/
Branch Lab Design Automation EAS
Zeunerstr. 38, D-01069 Dresden, Germany

Gerd Pönisch

Technische Universität Dresden
Institut für Numerische Mathematik
Dresden, Germany

Manfred Uhle

Weierstrass Institute for
Applied Analysis and Stochastics
Berlin, Germany

Abstract

Many circuits such as flip-flops, Schmitt triggers, and negative resistance circuits have multiple dc solutions. The same problem can occur in electrical non-electrical devices. For many years homotopy techniques have been investigated to obtain the multiple solutions. We present a method that establishes the resulting curve-tracing problem as a network analysis problem using the hardware description language VHDL-AMS. Thus, the curve-tracing problem can be solved with an available VHDL-AMS simulation engine. A special customized simulator is not necessary. The methodology can also be applied to the calculation of transfer characteristics and turning points. This allows e. g. the determination of pull-in voltages in MEMS.

Introduction

One fundamental task in circuit simulation is the determination of the dc operating point (see e. g. [1]). From a mathematical point of view a nonlinear system of equations has to be solved. Such systems have been under investigation for a long time until today. The interest results from the difficulties that may occur during the dc analysis. In some cases, more than one operating point exists. For instance, circuits as flip-flops, Schmitt-triggers and negative resistance circuits have multiple dc solutions. The state-of-the-art method to solve such problems is the application of continuation techniques based on homotopy methods. Usually special simulation algorithms have to be implemented to realize this approach. This is not necessary if the procedure that is suggested in the following will be executed.

The homotopy idea is directly applied to the network analysis problem. A network is constructed which is similar to the original one. From the solution of the augmented network we simply get the solution of the network we are interested in. The equation that realizes the continuation condition in the mathematical approach is kept by an additional subcircuit of the augmented network. By this way the augmented network can completely be expressed using a behavioral description language. We will discuss how this can be done using VHDL-AMS [2].

The advantage of this procedure is that no special simulation engine is necessary to investigate problems with multiple solutions. The approach can be extended in a natural way to the determination of multivalued dc transfer characteristics. The turning points of such characteristics define e. g. hysteresis parameters of Schmitt-trigger circuits. A similar problem has to be solved if pull-in parameters of MEMS devices have to be identified. If the MEMS problem can be described by lumped elements and subsystem models that can be implemented in a description language like VHDL-AMS, the same method can be used. Especially if VHDL-AMS is used, the pull-in parameters can be determined as a simulation result and not only in a post-processing examination.

At the beginning of the paper some results from homotopy analysis are recapitulated. Afterwards, the basic ideas to construct augmented networks that realize continuation techniques based on homotopy methods are introduced. A brief overview on the implementation of these ideas using VHDL-AMS follows. Some illustrative examples demonstrate results of the application of the suggested approach.

A lot of work has been done in the field of investigation of nonlinear systems having multiple solutions. Our method unifies a wide number of known approaches. The usage of a standard behavioral language like VHDL-AMS offers the opportunity to investigate the discussed class of problems in a simple way using available simulation engines. Last but not least we think that the presented relation between the curve tracing method and the pull-in phenomenon in micro electromechanical devices improves the possibilities of behavioral simulation in the MEMS design process..

Homotopy Methods for Nonlinear Systems of Equations

The dc operating point is the solution of a system of nonlinear equations $F(x) = 0$, $F : R^n \rightarrow R^n$. Continuation methods are widely used to solve such problems. They base on a homotopy function $H : R^n \times R \rightarrow R^n$. The function H is continuous differentiable with the following characteristics:

- The solution $x_0 \in R^n$ of $H(x_0, \lambda_0) = 0$ can be determined for a given $\lambda_0 \in R$.
- For a special $\lambda_f \in R$ the solution $x_f \in R^n$ of $H(x_f, \lambda_f) = 0$ is also a solution of F , i.e. $F(x_f) = 0$.

λ is called homotopy or continuation parameter. Examples for homotopy functions H with $\lambda_0 = 0$ and $\lambda_f = 1$ are [3], [4]

- $H(x, \lambda) = F(x) + (\lambda - 1) \cdot F(x_0)$ (Newton homotopy)
- $H(x, \lambda) = \lambda \cdot F(x) + (\lambda - 1) \cdot (x - x_0)$ (linear homotopy)
- $H(x, \lambda) = F(x) + (\lambda - 1) \cdot e^{-k\lambda} \cdot F(x_0)$

The solutions of $H(x, \lambda) = 0$ form a set $S \subset R^n \times R$. A parameter representation can be given for a subset $L \subset S$, i.e.

$$L = \{(x(t), \lambda(t)) \mid H(x(t), \lambda(t)) = 0 \quad \text{with} \quad t \in [t_{\min}, t_{\max}]\}^1.$$

For almost all choices of $x_0 \in R^n$ a smooth path starting at $(x_0, 0)$ and terminating at $(x_f, 1)$ exists in the case of a linear homotopy. Arc length methods try to follow the curve L from a starting point $(x_0 = x(t_{\min}), \lambda_0 = \lambda(t_{\min}))$ with $t_{\min} = 0$ until a point $(x(t_{\max}), \lambda(t_{\max}))$. Solutions of the original problems are given by points (x, λ) of L where $\lambda = \lambda_f = 1$.

¹ x is used as a symbol for an element of R^n and as a symbol of for a function with R^n values. The meaning follows from the context.

The solution curve L can be computed by a discrete set of solution points using curve tracking methods. For that purpose, the equation

$$H(x(t), \lambda(t)) = 0 \quad (1)$$

has usually to be extended by a further equation.

A. Monotonic Continuation

In the trivial case of monotonic continuation (1) has to be augmented by

$$\lambda(t) - \lambda_0 = t \quad (2).$$

A functional relationship is assumed between x and λ .

B. Pseudo Arc-Length Method I

Well known methods base on a pseudo arc-length approach. Then, equation (1) is extended by

$$r_a^T \cdot (x(t) - x_0) + r_b \cdot (\lambda(t) - \lambda_0) = t \quad \text{with } r_a \in R^n \text{ and } r_b \in R \quad (3)$$

where r_a and r_b have to be chosen in such a way that the bordered matrices $B(x, \lambda) := \begin{pmatrix} \partial_x H(x, \lambda) & \partial_\lambda H(x, \lambda) \\ r_a^T & r_b \end{pmatrix}$ are non

singular for all $(x, \lambda) \in L$. An optimal choice r_a and r_b minimizes the condition number of $B(x, \lambda)$, see [5]. In this case t is about the arc length of the solution curve. The equation (1) and its augmentation (3) can be solved by monotonic continuation of t from $t_{\min}=0$ to t_{\max} . The monotonic continuation from subsection A is a special case of this method ($r_a = 0$, $r_b = 1$).

C. Pseudo Arc-Length Method II

Good experiences were gained using the arc length of the solution curve as parameter [6]. The augmentation equation in this case is

$$\sum_{i=1}^n \left(\frac{d}{dt} p_{ri} x_i(t) \right)^2 + \left(\frac{d}{dt} \lambda(t) \right)^2 = 1 \quad (4)$$

where $p_{ri}x$ is the i th component of x . The equation (1) and its augmentation (4) form a differential algebraic system of equations. These kind of systems are solved in the transient analysis of Spice-like network analysis programs.

In practical realizations, it is necessary to fix the direction of the curve tracking at the starting point. Furthermore, not all components of x are taken into consideration in (4). Indices of essential variables form the set I . Thus, instead of (4), the following augmentation equations are applied ($t_{\min}=0$, $0 < t_a < t_b < t_{\max}$):

$$\bullet \quad t \in [0, t_a] : \quad \lambda - \lambda_0 - t = 0 \quad (5a)$$

$$\bullet \quad t \in [t_a, t_b] : \quad \left(\frac{d}{dt} \lambda(t) \right)^2 + \left(3 \cdot t_h^2 - 2 \cdot t_h^3 \right) \sum_{i \in I} \left(\frac{d}{dt} p_{ri} x_i(t) \right)^2 = 1 \quad (5b)$$

$$\text{where } t_h = \frac{t - t_a}{t_b - t_a}$$

$$\bullet \quad t \geq t_b : \quad \left(\frac{d}{dt} \lambda(t) \right)^2 + \sum_{i \in I} \left(\frac{d}{dt} p_{ri} x_i(t) \right)^2 = 1 \quad (5c)$$

The equations (1) and (5a)-(5c) form a differential algebraic system of equation that has to be solved. The set I has to be choosen so that the Jacobian of this DAE system is regular.

Network Formulation of Homotopy Methods

The direct application of homotopy methods allows to find

multiple operating points (see e. g. [4]). This normally requires some extensions in the code of a simulation engine. But the ideas of the previous section can be directly accepted for network simulation problems [1], [6]. We assume, a “difficult” dc simulation problem shall be solved for a *resistive* network N . A network is resistive if its solution only depends on the instantaneous values of voltage and current sources. At first, a “homotopy” network N_λ has to be constructed. This is in accordance with the formulation of a homotopy function (1). In a second step a subsystem that realizes the augmentation equation (either (2), (3), (4) or another one) has to be connected to the “homotopy” network. The solution of the interconnection gives a solution of the original task. An existing network simulation program can be used to find multiple solutions.

A. Construction of Networks N_λ Close-by the Original N

We construct a network N_λ . λ is an additional network variable. The constitutive relations of some network branches depend on λ . To solve N_λ , λ must be constrained by an additional equation. It is assumed that the solution of N_λ can easily be found for $\lambda = \lambda_0 = 0$. Under a special condition concerning λ , the solution of N_λ implies the required solution of the original network N . Known construction methods are denoted in the following (for references see [4]). The described subnetworks are controlled by an input voltage λ .

- Independent voltage and current sources of the original network N with values V_{cl} and I_{cl} resp. are replaced by sources with the values $\lambda \cdot V_{cl}$ and $\lambda \cdot I_{cl}$ resp. It is evident that for $\lambda = 1$ the solution of N_λ implies the solution of N (*source stepping*). The VHDL-AMS code of such an independent voltage source is given in principle below. The voltage λ at terminal t_{λ} controls the voltage between p and m .

```
entity vss is
  generic ( vc1 : real );
  port (terminal p, m : electrical;
        terminal t_lambda : electrical );
end entity vss;

architecture a1 of vss is
  quantity v across i through p to m;
  quantity lambda across t_lambda;
begin
  v == lambda*vc1;
end architecture a1;
```

- Another way is to add conductances with values $(1 - \lambda) \cdot G_{\min}$ between nodes of N and the associated reference node. For $\lambda = 1$ the solution of N_λ implies the solution of N (*G-min stepping*).
- The *artificial input current source stepping* method was recently introduced by Ushida et al. [1]. Additional current sources with values $(1 - \lambda) \cdot I_{c0}$ are included into N to construct N_λ . VHDL-AMS enables us to make the method a little bit more flexible. Instead of introducing current sources with values I_{c0} for $\lambda = 0$ we determine the starting point for curve tracing by adding voltage sources with values V_{c0} between the selected nodes of N . It often seems much more easier to guess reasonable values V_{c0} than I_{c0} . Curve tracing is then carried out with the current source as prior described. The current I_{c0} is equal

to the current in the voltage branch for the determination of the starting point. The determination of the starting point is done during `quiescent_domain` simulation in VHDL-AMS. Curve tracing will be done during `time_domain` simulation. The VHDL-AMS code of the modified stepping source is given below. The value of I_{c0} is saved in a real-valued signal.

```
entity css is
  generic  ( vc0           : real);
  port    (terminal p, m   : electrical;
           terminal t_lambda : electrical);
end entity css;

architecture a0 of css is
  signal  ic0 : real;
  quantity v   across i through p to m;
  quantity lambda across t_lambda;
begin
  process(domain) is begin
    ic0 <= i;
  end process;

  if domain = quiescent_domain use
    v  == vc0;
  else
    i  == (1.0-lambda)*ic0;
  end use;
end architecture a0;
```

One or more of the described subsystems can be included in a network N to construct a N_λ . All of them are controlled by the same λ . The solution for the original problem can be found in all cases if the special condition $\lambda=1$ is fulfilled.

In some cases another condition can be applied. Fig. 1 shows a subnetwork that can replace a short circuit in an original network N to construct N_λ .

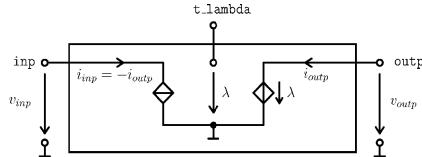


Fig. 1 Subnetwork that replaces a short branch of network N

Fig. 1 shows a subnetwork of N_λ that can be used instead of one short branch in N between inp and $outp$. A solution of the original network is given if $v_{inp} = v_{outp}$. The VHDL-AMS code can be found below.

```
entity transfer is
  port (terminal inp, outp : electrical;
        terminal t_lambda : electrical);
end entity transfer;

architecture a0 of transfer is
  quantity v_inp across i_inp through inp;
  quantity v_outp across i_outp through outp;
begin
  i_inp == -i_outp;
  v_outp == t_lambda'reference;
end architecture a0;
```

The subnet transfer can be used to break up one feedback loop in the original network N . The basic idea to construct a homotopy network not only for the special solution condition $\lambda=1$ but for another one was constituted by Reibiger et al [7]. Besides the network shown in Fig. 1 numerous other constructions are possible.

B. Networks for Curve Tracing

The dc solution can be found by connecting a network N_λ with a circuit that realizes one of the augmentation equations (2), (3)

or (5). The output of the “augmentation” network provides λ . The inputs of such a network are the components of x that have to be evaluated in the augmentation set. The overall network is shown in principle in Fig. 2.

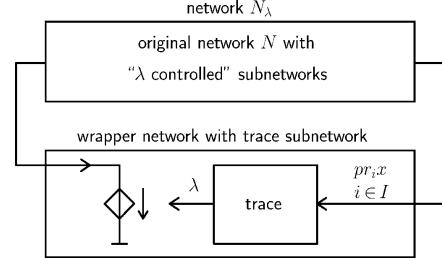


Fig. 2 Subnetwork that replaces a short branch of network N

The kernel of the network in addition to N_λ is the network $trace$. Its VHDL-AMS description does not depend on the network N_λ that is constructed for a special task. The VHDL-AMS model that realizes the second pseudo arc-length method is given in the following:

```
entity trace is
  generic  (N : natural := 1;
            ta : real   := 1.0e-3;
            tb : real   := 5.0e-3);
  port (quantity x      : in real_vector (1 to N);
        quantity lambda : out real;
        signal stop    : boolean := false);
begin
  assert ta > 0.0
    report "ta must be greater 0.0" severity error;
  assert tb > ta
    report "tb must be greater ta" severity error;
end entity trace;

architecture a1 of trace is
  quantity lambda_dot : real;
  quantity x_help     : real_vector (1 to N);
  quantity x_dot      : real_vector (1 to N);

  function "*" (a, b : real_vector) return real is
    variable result : real;
  begin
    result := 0.0;
    for i in a'range loop
      result := result + a(i)*b(i);
    end loop;
    return result;
  end function "*";

  function factor (t: real; ta : real; tb : real)
  return real is
    variable th     : real;
  begin
    th := (t-ta)/(tb-ta);
    return (3.0*th*th - 2.0*th*th*th);
  end function factor;

begin
  lambda_dot == lambda'dot;
  x_help    == x;
  x_dot     == x_help'dot;
  if domain = quiescent_domain use
    lambda == 0.0;
  else
    if not stop use
      if now < ta use
        lambda == now;
      elsif now < tb use
        1.0 == lambda_dot*lambda_dot
              + factor(now,ta,tb)*(x_dot*x_dot);
      else
        1.0 == lambda_dot*lambda_dot
              + x_dot*x_dot;
      end use;
    else
      lambda'dot == 0.0;
    end use;
  end use;
end architecture a1;
```

The trace network is a nonconservative network with quantity ports. The wrapper network depends on the special problem. It provides the components of vector x of network variables that are used in the augmentation set as input quantities of trace. The choice of the components of x depends on the original network N and its augmentation N_λ .

In principle all components of x can be selected. But this is not very suitable. The choice must be directed by the intention to assure the regularity of the Jacobian of the network equations during curve tracking. In transistor networks it seems to make sense to select some nodal voltages of feedback loops for this purpose [1], [8].

The output λ of trace controls a voltage source in the wrapper network. This voltage source is connected to the additional subnets in N_λ . The curve tracing is stopped if the input signal stop of the trace subnet is set to true.

The network shown in Fig. 2 can be evaluated with the time domain simulation algorithm of a VHDL-AMS engine. The starting point is evaluated during quiescent domain simulation.

Handling of Simulation Tasks using VHDL-AMS

Different simulation tasks can be carried out using the network arrangement from Fig. 2. By this way the solution curve of N_λ can be tracked.

A. Determination of Operating Points

To detect the operating point it must be checked whether the special condition is fulfilled that makes possible to conclude a dc solution of N from a solution point of N_λ . As discussed in part A. of the previous section this condition depends on the added subnets.

In many cases (*source stepping*, *G-min stepping*, ...) the condition $\lambda=1$ has to be detected. In VHDL-AMS this can be done using the `'above` attribute. The following VHDL-AMS code realizes such a detection. If the λ -voltage is applied to terminal `t_lambda` and a voltage we are interested in is connected to terminal `t_voltage` the signal `s_voltage` delivers the voltage at dc solution points.

```
entity detect is
port(t_lambdA, t_voltage : electrical;
      signal s_voltage : out real := 0.0;
      signal stop       : out boolean := false);
end entity detect;

architecture one of detect is
  quantity v_lambda across t_lambda;
  quantity v          across t_voltage;
begin
  process (v_lambda'above(1.0)) is
  begin
    s_voltage <= v;
  end process;
  stop <= false;
end architecture one;
```

B. Determination of Characteristics

The network from Fig. 2 can also be used to determine multivalued dc transfer characteristics. A typical characteristic of this class is a transfer characteristic of a Schmitt trigger circuit. The basic task is to determine the dependency of an output quantity (e. g. special voltage or current) on an input quantity (e. g. input voltage or current) in a resistive network N . Multivalued characteristics occur if there are branches of the characteristic where the input quantity increases and other branches where it decreases.

N is extended by a suitable input source in order to construct N_λ . The input source is controlled by λ . An example of such an input source is the independent voltage source `vss` from the previous section. Curve tracing is then done using an overall network as given by Fig. 2. The projection of the traced solution curve onto the input-output plane delivers the desired characteristic.

C. Determination of Turning Points

We assume that x is the vector of network variables of a N_λ . If the input quantity v which is a component of x does not increase (or decrease) monotonic on the way from a solution point (x_0, λ_0) to a solution point (x_m, λ_{\max}) then there exists a turning point. If the solution curve is parametrized by t , for example, then a turning point is characterized by a local extremum $v(t^*)$ (minimum or maximum, respectively) of the continuous function $v=v(t)$. Hence, if $v(t)$ is differentiable in an environment of t^* the following condition has to be fulfilled [9]

$$\operatorname{sgn}\left(\frac{dv}{dt}(t^* - 0)\right) = -\operatorname{sgn}\left(\frac{dv}{dt}(t^* + 0)\right) \quad (6).$$

This condition can be evaluated using the `'above` attribute in VHDL-AMS. This can be done using the following architecture `turning` of the entity `detect`. v is equal to the voltage at the terminal `t_voltage`. The voltages at turning points are assigned to the signal `s_voltage`. The turning point coordinates can be used to determine the values of a hysteresis.

```
architecture turning of detect is
  quantity v across t_voltage;
begin
  process is begin
    wait on domain;
    while true loop
      wait on v'dot'above(0.0);
      s_voltage <= v;
    end loop;
  end process;
  stop <= false;
end architecture turning;
```

Illustrative examples

In the following we will indicate how to solve typical benchmark problems [1], [4], [10] using the presented approach. The examples were carried out using the program ADVance MS [11]. On request, the authors can make the complete source codes available.

A. Two-Tunnel-Diode Circuits

The two-tunnel-diode circuit as shown in Fig. 3 is a well-known benchmark problem. The characteristics of the tunnel diodes are given by

$$i_1 = 2.5 \cdot v_1^3 - 10.5 \cdot v_1^2 + 11.8 \cdot v_1 \\ i_2 = 0.43 \cdot v_2^3 - 2.69 \cdot v_2^2 + 4.56 \cdot v_2$$

The artificial source stepping approach is used to construct a network N_λ . A `ccs` model with the initial voltage `vc0=0.0` is included between the nodes `n2` and the electrical reference node. The diagram in Fig. 4 gives as dc solutions for the voltage at node `n2` (values of signal `voltage_n2`) 1.05689, 1.89281, 3.95401, 5.48268 and 5.91777. To get further solutions the simulation has to be repeated with other initial values `vc0` of the `css` model.

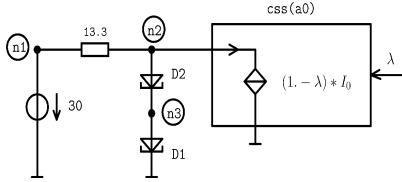


Fig. 3 Network N_λ for two-tunnel-diode circuit

The VHDL-AMS testbench describes the network from Fig. 3 with a wrapper (compare Fig. 2) for curve tracing and a subnet detect to detect the dc solutions.

```

entity bench is end entity bench;

architecture a1 of bench is
terminal n1, n2, n3 : electrical;
terminal lambda : electrical;
-- branch declarations
quantity vin across iin through n1;
quantity vr across ir through n1 to n2;
quantity v1 across il through n2 to n3;
quantity v2 across i2 through n3;
-- signal to describe dc solution
signal voltage_n2 : real := 0.0;
signal stop : boolean;
begin
  vin == 30.0;
  vr == 13.3*ir;
  i1 == 2.5 *v1**3-10.5 *v1**2+11.8 *v1;
  i2 == 0.43*v2**3- 2.69*v2**2+ 4.56*v2;

M1: entity css(a0)
  generic map (vc0 => 0.0)
  port map (n2, electrical_ref, lambda);

M2: entity wrapper(a1)
  port map (n2, lambda);

M3: entity detect(one)
  port map (lambda, n2, voltage_n2, stop);
end architecture a1;

```

The wrapper model realizes a link between the network that is shown in Fig. 3 and the model trace(a1) for curve tracing. The voltage at node n2 is the considered component of the vector of network variables.

```

entity wrapper is
  generic (ta : real := 1.0e-3;
           tb : real := 5.0e-3);
  port (terminal t_voltage, t_lambda : electrical);
end entity wrapper;

architecture a1 of wrapper is
  quantity vin across t_voltage;
  quantity v_lambda across i through t_lambda;
  quantity x : real_vector (1 to 1);
  quantity lambda : real;
  signal stop : boolean := false;
begin
  x(1) == vin;

M: entity trace(a1)
  generic map (ta => ta, tb => tb)
  port map (x, lambda, stop);

  v_lambda == lambda;

end architecture a1;

```

The results of the transient simulation are shown in Fig. 4. If λ equals 1 a dc solution of the two-tunnel-diode problem is investigated. Then, the actual value of the nodal voltage at n2 is assigned to the signal voltage_n2.

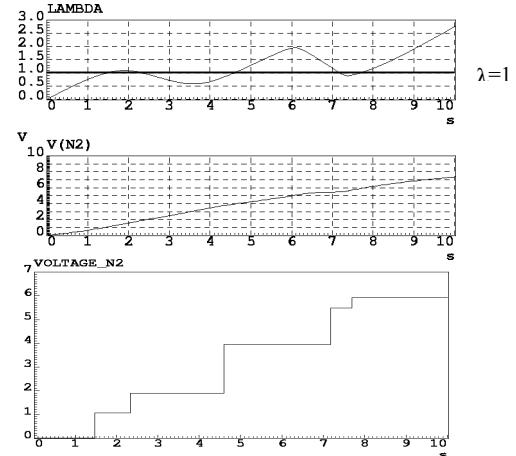


Fig. 4 dc solutions of node n2 for two-tunnel-diode network
(signal voltage_n2 represents the dc solution at node n2)

B. Hybrid Voltage Reference

The Banwell's hybrid voltage reference circuit is investigated in the same way. Fig.4 shows the constructed network N_λ . The initial voltage of css (a0) for the determination of the starting point was set to 1.8

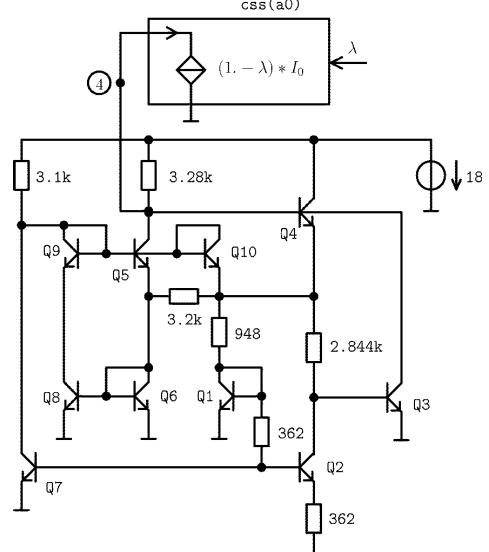


Fig. 5 Network N_λ for the hybrid voltage reference circuit
(default Spice parameters, area of Q2 and Q4 equals 3.33)

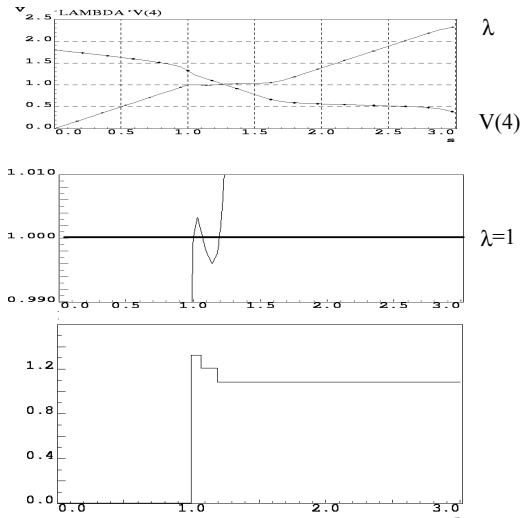


Fig. 6 dc solutions at node 4 for the hybrid voltage reference circuit
(1.32768, 1.20875, 1.08599)

C. Flip-Flop

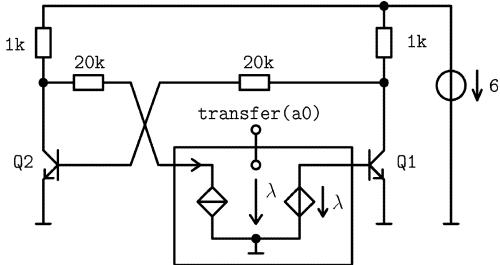


Fig. 7 Network N_λ for a flip-flop
(default Spice parameters, $\text{is}=1.0e-12$)

To determine the dc solution of the flip-flop that is shown in Fig. 7 the feedback structure is broken and the subnet from Fig. 1 is included. We get the three dc solutions for the collector voltages 0.0899, 1.47153 and 5.74208.

D. Schmitt-Trigger

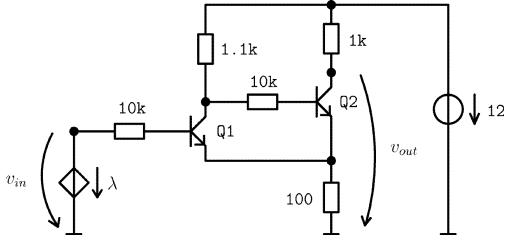


Fig. 8 Network N_λ for the calculation of a transfer characteristic
(default Spice parameters)

Fig. 8 shows the input circuit of the Schmitt-trigger cascade benchmark with a modified input voltage that is driven by the homotopy variable λ . Thus, the transfer characteristic can be calculated. The result is shown in Fig. 9. The coordinates of the turning points can be detected by including the design unit detect (turning) into the simulated circuit. Input voltages of the turning points are 2.72762 and 3.34416

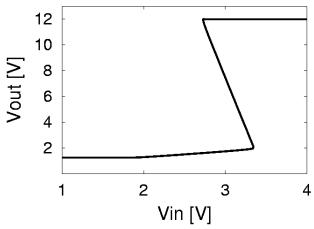


Fig. 8 Transfer characteristic

E. Parallel-Plate Actuator

The transfer characteristic is also helpful to investigate the pull-in phenomenon in electrostatic actuators [12]. Fig. 8 shows the typical problem. A voltage drives the electrostatic parallel plate actuator. The pull-in voltage limits the continuous travel range of the movable plate. For voltages, greater than the pull-in voltage, the plates come together. The pull-off voltage is smaller than the pull-in voltage and depends on the remaining gap between the gaps after pull-in. Pull-in and pull-off voltages are given by the turning points of the transfer characteristic [13]. Fig. 10 shows the result of the simulation for an ideal plate actuator. A more complicated example is given in [9].

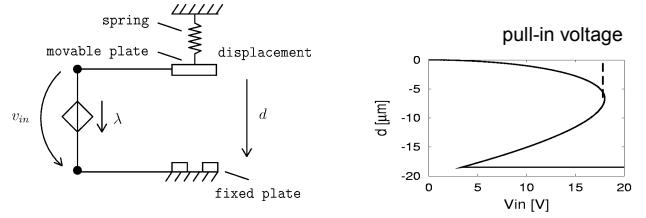


Fig. 10 Network N_λ for ideal parallel plate actuator and transfer characteristic

Conclusion and Remarks

Modern hardware description languages as VHDL-AMS offer the opportunity to formulate network analysis problems that realize systems of equations to track solution curves of homotopy problems. Thus, networks with multiple dc solutions, and dc multivalued transfer characteristics can be evaluated with existing standard simulation engines. Especially the 'above' attribute of VHDL-AMS provides a natural way to determine exactly dc solution points and turning points of transfer characteristics. The presented approach can be applied not only on pure electrical networks but also on electrical nonelectrical networks. This is interesting in the application of simulation for MEMS problems. The discussion of the stability of DC operating points (see [14]) is beyond the scope of this paper.

References

- [1] A. Ushida, Y. Yamagami, Y. Nishio, I. Kinouchi, and Y. Inoue, "An efficient algorithm for finding multiple DC solutions based on the SPICE-oriented Newton homotopy method," *IEEE Trans. Computer-Aided Design*, vol. 21, pp. 337-348, March 2002.
- [2] E. Christen and K. Bakalar, "VHDL-AMS – A hardware description language for analog and mixed-signal applications," *Trans. Circuits Syst.-II*, vol. 46, pp. 1263-1272, October 1999.
- [3] D. M. Wolf and S. R. Sanders, "Multiparameter homotopy methods for finding DC operating points of nonlinear circuits," *IEEE Trans. Circuits Syst.-I*, vol. 43, pp. 824-838, October 1996.
- [4] R. C. Melville, L. Trajkovic, S.-C. Fang, and L. T. Wayne, "Artificial parameter homotopy methods for the DC operating point problem," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 861-876, June 1993.
- [5] G. Pönnisch, "Two-stage methods for computing singular points of underdetermined nonlinear equations," *Z. Angew. Math. Mech.* vol. 74, pp. T664-T665, No. 6 (1994).
- [6] J. Haase, "Computation of transfer characteristics of multivalued resistive nonlinear networks," in *Short Contribution Summer School Circuit Theory SCCT'82*, Prague, 1982, pp. 268-272.
- [7] A. Reibiger, W. Mathis, T. Nähring, L. Kronenberg, and L. Trajkovic, "Mathematical foundation of the TC-method for computing multiple DC-operating points," in *Proc. ISET '01*, Linz, 2001.
- [8] R. O. Nielsen and A. N. Willson, "A fundamental result concerning the topology of transistor circuits with multiple equilibria," *Proc. IEEE*, vol. 68, pp. 196-208, Feb. 1980.
- [9] J. Haase and G. Pönnisch, "Determination of turning points in electromechanical systems," *Proc. 4th MATHMOD (vol. 2 full papers CD)*, Vienna, 2003, pp. 712-720.
- [10] Netlib benchmark library (<http://www.netlib.org/benchmark/biop/>)
- [11] ADVance MS Datasheet. Available: http://www.mentor.com/ams/datasheets/advancems_ds.pdf
- [12] H. C. Nathanson, W. E. Newell, R. A. Wickstrom, and J. R. Davis JR, "The resonant gate transistor," *IEEE Trans. Electro Devices*, vol. 14, pp. 117-133, No. 3(1967).
- [13] J. A. Pelesko, "Multiple solutions in electrostatic MEMS," *Proc. MSM 2001*, Hilton Head (SC), 2001, pp. 290-293.
- [14] M.M. Green and A.N. Willson, "An Algorithm for Identifying Unstable Operating Points Using SPICE", *IEEE Trans. Computer Aided-Design*, vol. 14, pp. 360-370, March 1995.