Computer-Aided Circuit Analysis with Respect to Switched Circuits

TOMÁŠ LUKL, VÍT NOVOTNÝ, JIŘÍ MIŠUREC
Department of Telecommunications
Brno University of Technology
Purkynova 118, 612 00 BRNO
CZECH REPUBLIC

Abstract: - This article deals with issues which we can meet when we want to analyze nonlinear dynamic circuits using Matlab environment. Theoretical basements of analysis of general type of circuits and their possible solution as computer algorithms are discussed first. Then methods of computer analysis of switched circuits are described later on.

Key-Words: - Computer analysis, Switched circuits, Matlab, Numerical methods.

1 Introduction

During nonlinear dynamic circuit analysis (analog or switched) we have to solve a lot of issues. First we have to choose a good method how to effectively (using a computer) analyze wide sort of circuit configurations and types. Modified Nodal Analysis (MNA) seems to be the best choice. It allows us to analyze circuit with voltage and current defined elements and simply compute any other circuit variables (such as charges and/or fluxes). We get a set of nonlinear differential equations when applying MNA method. We could find a solution of such set of equations in closed analytical form in some cases (e.g. linear dynamic circuits) but mostly it is not possible. So we have to touch an issue of numerical methods for solution of these equations. Nonlinear set of equations is transformed into linear algebraic one using processes called time discretization (sometimes called numerical integration) and linearization. These processes forms loop-in-loop algorithm, where inside loop performs linearization (it solves discretizied nonlinear differential set of equations from outside loop).

Analysis of switched circuits (also dynamic circuits) brings other issues we have to study. There is necessary to use charges and fluxes as circuit variables in analysis methods because these variables acquire finite values in switching times (instead currents and voltages). We also have to compute exact switching time (when internally controlled switches are used) and values of all necessary circuit variables in this time. These special issues in switched circuit analysis are the main reasons why to study and form new and enhanced algorithms for circuit analysis.

2 Modified Nodal Analysis

Modified nodal analysis (MNA) is general and often used method for circuit analysis using computer. MNA is the method for circuit equations formulation. There are also other more or less special methods such as Two-Graph [1] or Tableau [2] methods. We can see MNA general definition [2] in system of nonlinear differential equations (1) - (3).

$$\mathbf{A}_{\mathrm{I}}\mathbf{i}_{\mathrm{I}} + \mathbf{A}_{\mathrm{II}}\mathbf{i}_{\mathrm{II}} = \mathbf{0}\,, \tag{1}$$

$$\mathbf{i}_{\mathrm{I}} = \mathbf{f}_{\mathrm{I}}\left(\mathbf{A}^{\mathrm{t}}\mathbf{v}_n, \frac{d\mathbf{q}_{\mathrm{I}}\left(\mathbf{A}^{\mathrm{t}}\mathbf{v}_n\right)}{dt}, \mathbf{i}_{\mathrm{II}}, \frac{d\boldsymbol{\psi}_{\mathrm{I}}\left(\mathbf{i}_{\mathrm{II}}\right)}{dx}, \mathbf{s}(t), t\right), \tag{2}$$

$$\mathbf{f}_{\mathrm{II}}\left(\mathbf{A}^{\mathrm{t}}\mathbf{v}_n, \frac{d\mathbf{q}_{\mathrm{II}}\left(\mathbf{A}^{\mathrm{t}}\mathbf{v}_n\right)}{dt}, \mathbf{i}_{\mathrm{II}}, \frac{d\boldsymbol{\psi}_{\mathrm{II}}\left(\mathbf{i}_{\mathrm{II}}\right)}{dt}, \mathbf{s}(t), t\right) = \mathbf{0}\,, \tag{3}$$

where $\mathbf{A}$ is an incidence matrix, $\mathbf{v}_n$ is a vector of unknowns nodal voltages, $\mathbf{s}$ is a vector of excitations, variables with index I are variables of current-defined branch (defined by equation $i = f\left(\mathbf{v}, \mathbf{i}_{\mathrm{II}}\right)$) and variables with index II are variables of voltage-defined branch and branch currents, which are to be considered in analysis (we want to compute them). Equation set (1) - (3) is used for analysis of wide sort of nonlinear dynamic circuits.

When analyzed using computer we finally solve a system of linear equations

$$\mathbf{G} \cdot \mathbf{x} = \mathbf{y}\,, \tag{4}$$

where matrix $\mathbf{G}$ describes how elements are interconnected and how they are defined, $\mathbf{x}$ is a vector of unknown circuit variables (such as currents, voltages, charges and fluxes) and $\mathbf{y}$ is a vector of excitations. The problem is how to fill in the matrix $\mathbf{G}$ and the vector $\mathbf{y}$. Most simple and quick way is to use stamp technique [4]. Every single stamp describes an element. Stamps can be obtained directly from an element model. This way can be used for linear static elements only. Form of stamps for other types of elements depends on used linearization and discretization methods. Then, the matrix $\mathbf{G}$ and the vector $\mathbf{y}$ is a sum of stamps of all elements according to the way how they are interconnected. To solve the system of linear algebraic equations (4) is. the final phase of simulation. We have to choose fast and efficient method how to solve this system, because this block of algorithm is placed in the inside loop, also it is executed most times. We can use two types of methods:

- Finite methods – exact solution can be obtained in a finite number of operations. Best known is for example LU factorization or Gaussian elimination. Last one is used in Matlab command $\mathbf{x} = \mathbf{G} \backslash \mathbf{y}$;
- Relaxation methods – solution is obtained in (theoretically) infinite number of operations using an iteration cycle.

3 Methods and Algorithms

3.1 Linear Circuits

This is the simplest kind of circuits. When we want to understand the principles of circuit analysis and to form algorithms for computer aided circuit analysis, first we have to begin with these types of circuit with rather demonstration importance. MNA equations change as follows

$$\begin{bmatrix} \mathbf{A}_{\mathrm{I}}\mathbf{G}_{\mathrm{I}}\mathbf{A}^{\mathrm{t}} & \mathbf{A}_{\mathrm{I}}\mathbf{K}_{\mathrm{I}} + \mathbf{A}_{\mathrm{II}} \\ \mathbf{L}_{\mathrm{II}}\mathbf{A}^{\mathrm{t}} & \mathbf{K}_{\mathrm{II}} \end{bmatrix} \begin{bmatrix} \mathbf{v}_n \\ \mathbf{i}_{\mathrm{II}} \end{bmatrix} = \begin{bmatrix} -\mathbf{A}_{\mathrm{I}}\mathbf{I}_{\mathrm{I}} \\ \mathbf{s}_{\mathrm{II}} \end{bmatrix}, \tag{5}$$

where $\mathbf{G}$ is a conductance matrix, $\mathbf{K}$ and $\mathbf{L}$ are coefficients of controlled current and voltage sources in each branch and $\mathbf{I}$, $\mathbf{s}$ are excitations vectors. LHS (Left Hand Side) matrix is sum of stamps as mentioned above. Such stamp can look like the one shown in Table 1.

	1	2	i_e	
1			1	
2			-1	
i_e	1	-1		U

Table 1: Stamp matrix for d.c. voltage source U. Double line in the table divides excitation part (RHS) and conductance part (LHS), i_e is the current through the element.

3.2 Nonlinear Circuits

We solve this type of circuits during d.c. analysis (direct current), where capacitors are replaced by open circuit and inductors by short circuit. We can use following algorithms during a.c. analysis, but excitations must vary in time very slowly. We get MNA equations (1) - (3) for this type of circuits by zeroing all derivatives, see eq. (6) - (8).

$$\mathbf{A}_{\mathrm{I}}\mathbf{i}_{\mathrm{I}} + \mathbf{A}_{\mathrm{II}}\mathbf{i}_{\mathrm{II}} = \mathbf{0}, \tag{6}$$

$$\mathbf{i}_{\mathrm{I}} = \mathbf{f}_{\mathrm{I}}\left(\mathbf{A}^{\mathrm{t}}\mathbf{v}_n, \mathbf{0}, \mathbf{i}_{\mathrm{II}}, \mathbf{0}, \mathbf{s}(t), t\right), \tag{7}$$

$$\mathbf{f}_{\mathrm{II}}\left(\mathbf{A}^{\mathrm{t}}\mathbf{v}_n, \mathbf{0}, \mathbf{i}_{\mathrm{II}}, \mathbf{0}, \mathbf{s}(t), t\right). \tag{8}$$

System of equations (6) - (8) is generally nonlinear. It is difficult to solve even a simple nonlinear equation, also we have to use some numerical method to solve the system (6) - (8). These methods are based on an iteration cycle, in which we approximate the exact solution. Newton-Raphson’s method is widely used for solution of nonlinear equations in form

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}. \tag{9}$$

The principle of the method is defined by eq. (10), which is derived from Taylor’s series expansion of equation (9).

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \left[\mathbf{f}'\left(\mathbf{x}^{(k)}\right)\right]^{-1} \cdot \mathbf{f}\left(\mathbf{x}^{(k)}\right), \tag{10}$$

where k is number of the current iteration cycle. For better computational efficiency (more simple RHS of system (6) - (8)) we write eq. (10) in form of incremental linearization equation (11) - (12).

$$\mathbf{f}'\left(\mathbf{x}^{(k)}\right) \cdot \Delta\mathbf{x}^{(k+1)} = -\mathbf{f}\left(\mathbf{x}^{(k)}\right), \tag{11}$$

$$\Delta\mathbf{x}^{(k+1)} = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}. \tag{12}$$

These iterations need not converge. Here in following few paragraphs we can see some of the reasons and their possible solutions.

Nonlinear function $f(x)$ has exponential character and that’s why overflow or underflow of processor’s registers may occur. We usually replace exponential functions with their piecewise approximation. Equation (13) shows one of possible approximation [2].

$$\tilde{e}^x = \begin{cases} e^{40}(x-39) & x > 40 \\ e^x & 0 \le x \le 40 \\ \dfrac{1}{1-x} & x < 0 \end{cases}. \tag{13}$$

Oscillation or divergence of iterations may occur in some cases of $\mathbf{f}(\mathbf{x})$ waveform and initial point $\mathbf{x}^{(0)}$ selection. We use step limiting algorithms, which can prevent divergence or oscillation. Equation (14) shows step limiting algorithm developed by Agnew [2]. It is used in simulation core of testing version of program SCISIP [3].

$$\Delta\tilde{x}^{(k+1)} = \mathrm{sign}\left(\Delta x^{(k+1)}\right)\min\left\{\left|\Delta x^{(k+1)}\right|, \alpha^{(k+1)}\left|\Delta\tilde{x}^{(k)}\right|\right\}, \quad (14)$$

where $\Delta\tilde{x}$ is corrected increment and dumping factor $\alpha^{(k+1)}$ can be for example defined as follows

$$\alpha^{(k+1)} = \begin{cases} 1.5 & \mathrm{sign}\left(\Delta x^{(k+1)}\right)\mathrm{sign}\left(\Delta\tilde{x}^{(k)}\right) > 0 \\ 0.5 & \mathrm{sign}\left(\Delta x^{(k+1)}\right)\mathrm{sign}\left(\Delta\tilde{x}^{(k)}\right) < 0\,. \\ 0.05 & k = 0 \end{cases} \quad (15)$$

Step limiting algorithm can prevent divergence or oscillation and can decrease number of iterations.
We can use several ways how to recognize, if convergence was reached (for iterations interruption). The most simple is to determine the difference between values of variables from two last iteration steps. A condition to break the iteration can be of form (16).

$$\begin{aligned} &\boldsymbol{\varepsilon}^{(k+1)} = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}, \\ &\max\left(\left|\boldsymbol{\varepsilon}^{(k+1)}\right|\right) \le \varepsilon_{\max}, \end{aligned} \quad (16)$$

where $\varepsilon_{\max}$ is maximum allowed deviation of solution, which is usually set by an user. In this simple case, there can be a problem with convergence, when we are analyzing circuit with PN junctions. When voltage across a PN junction in two adjacent iteration cycles get into reverse region of PN v-i characteristics, $\varepsilon^{(k+1)}$ can reach very small values and it can cause the break of the iterations, even if PN junction should be in open state. We can avoid this for example by setting $\varepsilon_{\max} < 0.1 I_s$.
Finally we solve a system of linear algebraic equations (17) $k_{\max}$ times (where $k_{\max}$ is maximum number of iterations) or until convergence is reached.

$$\mathbf{G}^{(k)}\mathbf{x}^{(k+1)} = \mathbf{y}^{(k)}. \quad (17)$$

Matrix $\mathbf{G}^{(k)}\left|\mathbf{y}^{(k)}\right.$ can be filled in using stamp technique. But these stamps can not be obtained from elements model, but must be created in simulation core itself. There exists simple algorithm how to do this. Models just have to provide functions of all model variables, and their derivatives with respect to these variables. For example, a model of static diode gets this output:

$$i_d = I_s\left(e^{\Theta u_d} - 1\right), \quad (18)$$

$$\frac{di_d}{du_d} = i_d' = \Theta I_s e^{\Theta u_d}. \quad (19)$$

Algorithm then forms stamp for this diode model (see Table 2).

	a	k	
a	$i_{\mathrm{d}}'\left(u^{(k)}\right)$	$-i_{\mathrm{d}}'\left(u^{(k)}\right)$	$-i_{\mathrm{d}}\left(u^{(k)}\right)$
k	$-i_{\mathrm{d}}'\left(u^{(k)}\right)$	$i_{\mathrm{d}}'\left(u^{(k)}\right)$	$i_{\mathrm{d}}\left(u^{(k)}\right)$

Table 2: Incrementally linearized stamp for static diode.

3.3 General Nonlinear Dynamic Circuits

For description of this type of circuits we use eqs. (1) - (3). These equations are used for transient analysis. But how we can see, we must deal with a system of nonlinear differential equations. First logical step is to eliminate derivatives from this system. We use numerical integration methods, so we get a system of nonlinear equations (20), which are in fact eqs. (1) - (3) written in more compact form.

$$\mathbf{F}\left[\mathbf{x}_n, \dot{\mathbf{z}}\left(\mathbf{x}_n\right), t_n\right] = \mathbf{0}, \quad (20)$$

where $\mathbf{x}_n$ is a vector of unknown circuit variables and vector $\dot{\mathbf{z}}\left(\mathbf{x}_n\right) = \dot{\mathbf{z}}_n$ is a vector of time derivatives of state variables.
Numerical integration means dividing continuous time scale into sequence of discrete points $\{t_n\}$. Neighbor point distance is called integration step $h = t_{n+1} - t_n$. We can write general equation for numerical integration as follows:

$$\dot{\mathbf{z}}_n = \gamma_0\left(\mathbf{z}_n - \mathbf{d}_z\right), \quad (21)$$

where γ_0 and $\mathbf{d}_z$ are coefficients which depend on numerical integration method and values from previous time steps.
Most used methods are Trapezoidal, Backward Differentiation Formulae (BDF) and Predictor-Corrector methods. Equation (22) expresses differentiation formula for BDF of 2$^{\mathrm{nd}}$ order.

$$\dot{z}_n = \gamma_0\left(z_n - d_n\right) = \frac{3}{2h}\left(z_n - \frac{4}{3}z_{n-1} + \frac{1}{3}z_{n-2}\right). \quad (22)$$

As discussed in [5], there is better to use charges-fluxes as state variables due to the reason of solution accuracy and stability. E.g. general nonlinear charge-defined branch $i = \dot{q}(u)$ has, after discretization and incremental linearization following, form

$$i_n^{(k+1)} = \gamma_0\left(\frac{\delta q}{\delta u}\right)_n^{(k)}\Delta u_n^{(k+1)} + \gamma_0\left[q\left(u_n^{(k)}\right) - d_q\right]. \quad (23)$$

If we use voltage-defined expression $i = C(u)\dot{u}$, we get more complicated formulae with derivatives of capacity, which can lead to inaccurate simulation.
Other issues, which have to be solved during time-domain analysis, are briefly described in following two paragraphs.

- We have to determine Local Truncation Error (LTE), which is a discrepancy between computed signal and its true counterpart.
- The need to control the time step h during analysis can be easily explained. When circuit response is sharp, there is need for short integration step. In the other side, when circuit response is smooth, integration step can be longer.

Finally we solve a system of linear algebraic equations again:

$$\mathbf{G}_n^k \mathbf{x}_n^{k+1} = \mathbf{y}_n^k . \tag{24}$$

The problem how to fill in matrix $\mathbf{G}_n^k \middle| \mathbf{y}_n^k$ is the same like the one, described in the chapter 3.2. When stamp technique is used, these stamps must be formed in simulation core itself. Model of each used element must contain necessary functions and their derivatives with respect to all their variables.

3.4 Switched Circuits

Switched circuits are special cases of nonlinear dynamic circuits. There are elements such as ideal switch (we can also form its stamp), which is in fact an ideal representation of a real switch like a MOSFET. Algorithm is very similar to the algorithm for nonlinear dynamic circuit analysis with some differences [4]. E.g. for externally controlled switches, when switching time occurs ($t_n = t_i$), circuit configuration is changed. The values of circuit variables from the time close before time t_i ($t_n = t_{i^-}$) are used as initial values for solution close after time t_i ($t_n = t_{i^+}$). When internally controlled switches are used, the situation is more complicated [5].

4 Applications

Methods and algorithms mentioned above are being implemented in complex simulation program for analysis of general switched circuits, called SCISIP (Switched CIrcuits SImulation Program). Matlab r.14 is used as development environment. Application has two main parts:

- Simulation core, which fully takes advantages of matrix based core of the Matlab tool. The algorithms of simulation core are based on the theory mentioned above.
- GUI – Graphical User Interface is used for high efficient, comfortable and transparent usage of the program.

The program consists of four modules. First two modules (called Symbol Editor and Model Designer) are used for design of electrical elements. We draw circuits schema using third module called Schematics Editor. The GUI of this module can be seen in Fig. 1. There a nonlinear switched circuit is drawn, compiled just for presentation purposes.

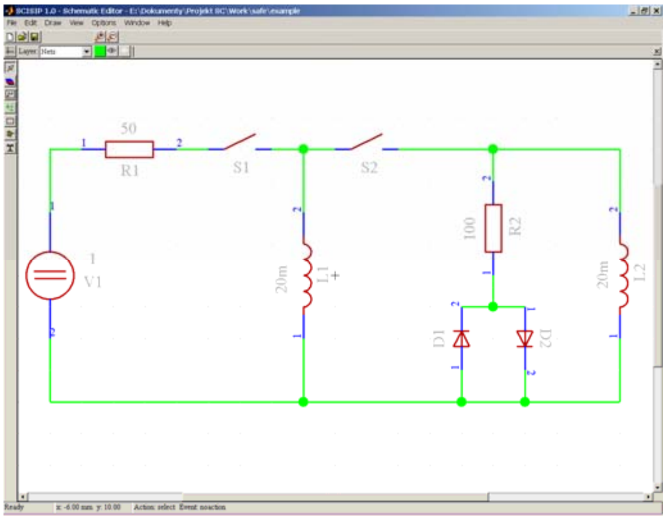

Fig. 1: A nonlinear switched circuit drawn in Schematic Editor module.

Last module, Simulation Manager, is used for setting up and executing a given analysis. This module displays simulations results as well. An example of graphical output of a simulation can bee seen in Fig. 2. Here, some currents in the circuit are drawn in the *Graph2* (green curve - i_{L1}, red curve - i_{S2}) and some voltages in the *Graph1* (green curve - u_{V1}, blue curve - u_5).
Program SCISIP (its simulation core and GUI) is still being developed; also presented examples are just draft versions. Programming a graphical application using Matlab tool has two big disadvantages:

- We have to own Matlab software and it has to be executed when we want to run our application. We can bypass this by compiling our Matlab code using *Matlab Compiler*, so we get a platform specific executable.
- Programming and running a GUI in the Matlab is not so fast and efficient as we would use another programming language such as Java or C++.

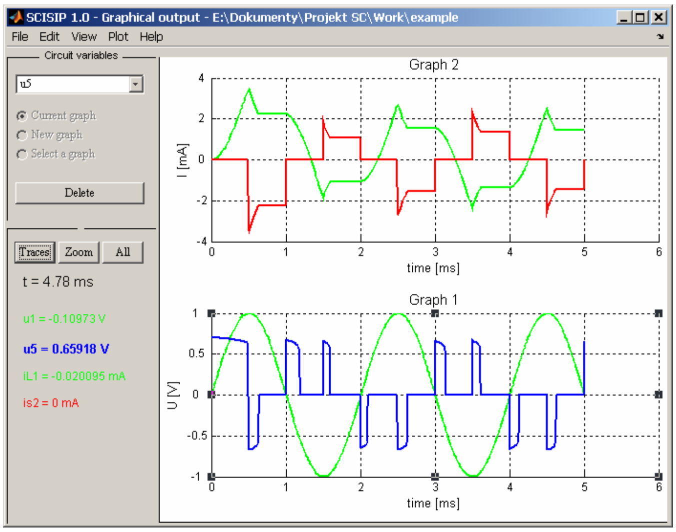

Fig. 2: SCISIP - graphical output after the time-domain analysis of nonlinear switched circuit from fig. 1.

5 Conclusion

This paper presents the main issues in switched circuits analysis by a computer. When we want to use simulation algorithms in praxis, we have to weight the options of increasing robustness of used iteration methods (enlarge reliability of convergence and solvability). In MNA method we have to strike wide sort of special types and configurations of circuits, e.g. isolated nodes and other special issues.

References:

[1] Horák, M., Kolka, Z., Simplification Before Generation using the Two-Graph Method, *EDS Conference 2004 Proceedings*, 2004, pp. 23-26.

[2] Ogrodzki, J., *Circuit Simulation Methods and Algorithms*, CRC Press, 1994, ISBN 0-8493-7894-X.

[3] Lukl, T., Novotný, V., Šponar, R., Neumann, J., Software for Switched Circuit Simulation (in Czech). *Elektrorevue – Internet journal (http://www.elektrorevue.cz)*, ISSN 1213-1539, 2003, no. 12, p. 2 - 11.

[4] Mann, H., Using computer in electrical design (in Czech), SNTL, Prague, 1984.

[5] Novotný, V., Lukl, T., Algorithms for Non-Linear Switched Network Simulation, *TSP 26th International Conference 2003*, 2003, pp. 205-208.

Acknowledgements:

This paper was worked out under the support of the projects FRVS 2466/2005/G1 and AVCR No. 1ET110530523.