Framework for Automated Worst-Case Analysis

Pál Weisz, György Orosz
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {weisz, orosz}@mit.bme.hu

***Abstract*—In this paper, a framework along with an electronic circuit database is presented. The framework can be used to test advanced worst-case circuit analysis algorithms. It has two main components: a software environment and a circuit database.**

The software environment is capable of running the circuit simulations and returning the results. The framework also provides a software interface between the circuit simulator and the data processing algorithms by transferring and converting data between different software components. Finally, the simulation results can be post-processed at different levels in time and frequency domains.

The heart of the framework is the circuit database. The database is intended to be used as a benchmark and test environment for advanced analysis algorithms. The circuits are collected and categorized to support promising research directions. Such research fields are: automatic decomposition of electrical circuits into independent or loosely coupled subcircuits; extreme value search in the case of complex circuits; identification of characteristic behavior of circuits where the system has different operation modes or some parameter constellations can result in different behavior.

***Index Terms*—worst-case analysis, test environment, circuit model, schematic, framework, simulation**

I. Introduction

For safety-critical systems, incompatibility with the required specifications can lead to accidents and environmental damage, so such applications are designed and verified with great care in the industry. The design is based on methods that comply with various standards and specifications. During this stage, different test cases are defined for the verification process. With the help of testing and verification, the system's performance becomes measurable.

It is more cost-effective and less hazardous to first test a model of the circuit in a simulated environment on a computer. A physical prototype is only realized after all the required test cases are satisfied.

When assessing compliance with requirements, it is important to know how a circuit's operation depends on the electrical components' parameters, different environmental influences, and failure modes. The analysis of the effect of extreme conditions on the circuit operation is called worst-case analysis (WCA) [1] [2]. In this type of test, the subjects of interest are extreme phenomena and their causes, where the circuit operation deviates as much as possible from the specification. Since a complex circuit may contain several hundreds or thousands of analysis tasks, the efficiency of the WCA solution is crucial.

WCA is typically performed using circuit simulation software (e.g. LTspice, OrCAD, Tina). These software tools provide some basic and traditional analysis methods for performing WCA, e.g. Extreme Value Analysis (EVA), Monte Carlo Analysis, and sensitivity analysis [1].

However, these programs do not always provide a fast and efficient solution. Simulator programs numerically solve system equations to provide solutions for various systems. Sometimes, it may be possible to describe a problem in analytical form and solve it more quickly than with simulators using numerical algorithms [3].

Several methods have been proposed in recent decades to solve WCA tasks efficiently. Some examples are interval arithmetic [4] or affine arithmetic [5] [6]. If the system is described in analytical form, it also has the potential advantage of applying advanced extreme value search techniques [3] [7] [8] [9].

The above discussion shows that there is no standard method in the worst-case analysis process and that there are several open issues. Promising research fields could be, for example, the design of intelligent analysis algorithms that involve the characteristic properties of electronic circuits as a priori knowledge, providing more explainable and interpretable results that can support the root cause finding when a requirement fails. In order to support this research, a framework is suggested in this paper capable of running and evaluating electronic circuit simulations and contains benchmark and test circuits.

The paper is structured as follows. Section II provides an overview of the motivation behind the importance of the proposed framework. Section III presents the functional architecture and the software environment's main components, as well as the design considerations behind them. Section IV introduces some examples from the circuit database suitable for testing and further research. These highlighted examples also serve to illustrate typical test methods.

II. Objective

The proposed framework was motivated by the needs of industrial practice. Experience has shown that moving the data generated during the analysis process between the different analysis and design environments is cumbersome. Not only is it necessary to manually move data from one computer program to another, but in many cases, it also takes extra time to convert the data formats. This is necessary because several development and test software environments cannot work effectively in a cascade.

Fig. 1 shows the role of the framework in the analysis process along with the different related components: circuits, computational tools, and higher-level algorithms.

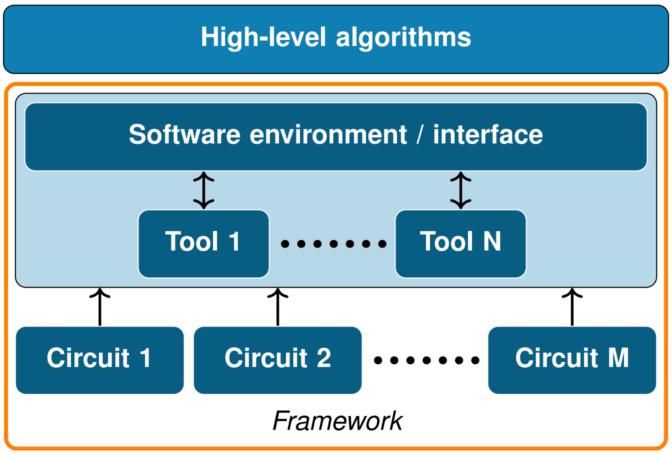


Fig. 1. Parts of the framework

Finding the least favorable behavior of a circuit is a challenging task, especially for more complex circuits, since several factors and parameters affect the circuit simultaneously. Many design and development steps precede the worst-case analysis. The process itself can be quite complex and time-consuming.

The proposed framework aims to optimize the entire worst-case analysis process and make it more efficient. On the other hand, it provides a consistent interface to output and input channels to support automation and greater flexibility in the different tools used for the analysis process.

III. Software environment

One of the fundamental elements of the framework is a software environment. The proposed software environment can produce the results of the circuit model evaluation. There are several options in order to do this.

In order not to be limited to the processing of simulation results, the software environment provides a uniform interface to the completely different methods that produce the results of the models.

A. General Structure

The overall architecture of the proposed software environment is shown in Fig. 2.

The software is built architecturally in layers. The essential task of the bottom layer is to evaluate the circuit model by simulation or other computational methods. This can be done by calling simulator programs or various solver algorithms that provide results based on the circuit model.

In the next layer, the software environment provides an interface between the circuit model solver/simulator and the data processing system, which can transfer and convert data between each part. This interface allows the software to read the data and also to parameterize the model circuit in an automated way.

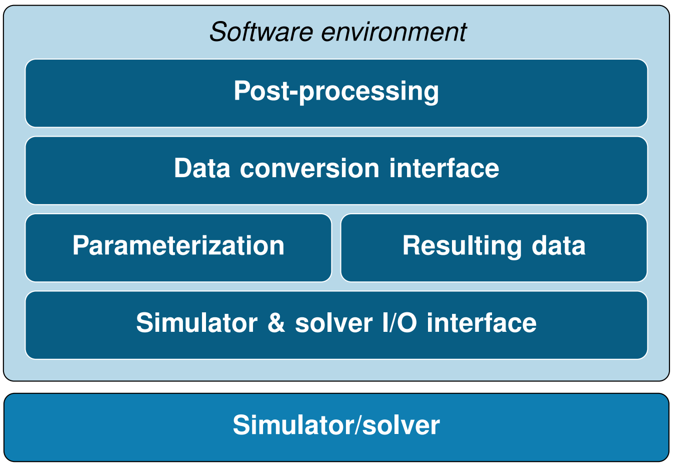


Fig. 2. Software Architecture.

The results are produced in a format depending on the method used. The resulting data must, therefore, be converted into a format that the software can handle consistently.

Finally, the data obtained can be post-processed at different levels in time and frequency domains. The proposed software environment can be used to test various high-level analysis methods.

The framework uses LTspice as a circuit simulator and MATLAB for mathematical modeling and further computations. These choices are sufficiently versatile that they do not limit the overall usability of the framework.

B. Functional Description

The low-level input-output interface has basic functionalities: modifying model parameters, starting the simulation, and retrieving results. In this particular case, the responsibility of the simulator I/O interface is to parameterize the circuit model and run the simulation by invoking LTspice commands.

The first important improvement presented in this paper is the choice of the way the parameters are set. As shown in Fig. 3, there are three ways of running the simulations for more than one parameter set:

- simulator program is started for every single parameter setting: general method but slow;
- simulator program is started for a batch of parameter set: more efficient, but requires parameter values in the actual batch in advance;
- analytical equations are generated from the schematic and parameters are substituted into equations: most efficient, but analytical form does not exist for each schematic.

When an iterative method of a worst-case analysis is being performed, there is a serious overhead in runtime to repeatedly restart the simulation. In order to mitigate this impact, the root causes need to be understood.

One reason for the runtime overhead is that the simulation software must be started (highlighted in bold font in Fig. 3)

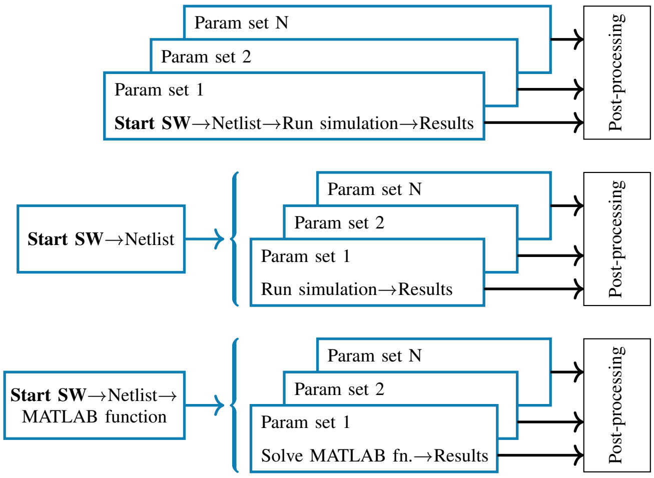


Fig. 3. Different approaches to computation methods

which produces a SPICE netlist. Generally, parameters are defined as constants in the circuit model's schematic diagram. The other option is to read the parameters from an external file. During simulation in LTspice, as a first step, the schematic diagram is transformed into a netlist. The simulation is being performed using this standard netlist file.

The simulation runtime can thus be reduced by omitting this transformation step and introducing the parameters only after the netlist has been generated (as shown in the second row of Fig. 3). From a less technical aspect, on the other hand, in some cases, it is possible to introduce multiple parameters at the same time to perform repeated analysis.

This means the simulation must be started less often to reduce the related overhead. However, suppose the parameters need to be changed frequently (e.g. to perform gradient-based extreme value finding, which the simulator cannot). In that case, it is advisable to manipulate the netlist in contrast with reading from an external file or change parameters in the schematic. While extra file operations are required in the first case, the netlist generation cannot be avoided in the latter case. If any option is chosen, an increase in runtime is expected.

In certain cases, it is possible to create an analytical representation of a circuit model [3]. If this option is available, excluding the simulation step from the process is possible.

The data conversion interface extracts the data from the simulation output files. It also selects their relevant records and converts them to the required format for higher-level use. The simulator calculates the results of all variables, but not all data is required. From the resulting data series, the ones that are relevant for data processing must be selected.

The top layer of the architecture implements the post-processing of the data. At this point, data in a format suitable for high-level analysis processes is available. Post-processing can be directly integrated with the worst-case analysis methods mentioned in Section I. At this point, the traditional mathematical apparatus can be applied.

IV. Circuit Database

The purpose of the circuit database is to support promising research fields in worst-case circuit analysis. Even advanced methods apply standard analysis steps, simple statistical methods, or blind evaluation of system equations without considering the electronic circuits' properties. In order to develop intelligent analysis algorithms, several test cases, illustrative examples, and benchmark circuits are needed.

The analysis can be performed more efficiently if the characteristic properties of the circuit are taken into account, as in a human analysis.

One of the suggestions for an intelligent analysis is that it is important to recognize the characteristic behaviors of the circuits. Some illustrative examples are distinguishing between operation modes (e.g. saturation, normal mode), separation of different behaviors in the frequency domain (e.g. high-pass, low-pass band, or different slope), identifying phases of a time-domain signal (e.g. rise-time, ringing, steady-state), or checking disjoint parameter constellations which can result in different operation modes (e.g. some load capacitance can result in the oscillation of an amplifier).

Decomposing the system into independent or loosely coupled subsystems could also be advantageous because it reduces the complexity of solving problems. So far, decomposition is done mainly based on human decisions, but it could be prone to error or influenced by subjective decisions. Hence, this paper also presents examples of where decomposition algorithms can be tested.

Traditional worst-case analysis algorithms often consider the linear approximation of the error surface [1]. Computationally efficient algorithms generally use only analytical formulas and could provide too conservative extreme value [4]–[6]. Numerical methods are promising [10], but there are several choices, and choosing the most effective is not trivial. So, finding the extreme value in complex systems is not trivial. Hence, problems with nontrivial extreme value are important, and such examples are also enumerated in the proposed database [11].

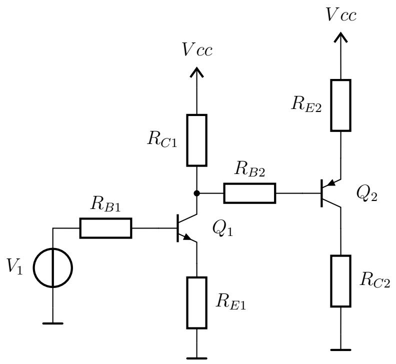
The electronic circuit database consists of different kinds of circuits based on real-life examples on which different analysis methods can be performed. Its schematic diagram defines a particular circuit model. A computational or simulation model of the circuit is used during the analysis.

In order to test different characteristic behaviors of a particular model, it is important to parameterize the circuits correctly to ensure that the desired properties are emphasized.

Some circuits from the database are listed below. An extended list of circuits can be found in [11]. A schematic diagram of the circuit and a representative simulation result for each example are presented. The simulation results are a qualitative representation of the circuit behavior under investigation, with a particular setting of the component parameters.

The first example is a switching circuit consisting of two bipolar junction transistors.

The schematic diagram and the simulation result are shown in Fig. 4.



V_1	R_{C1}	R_{B1}	R_{E1}	R_{C2}	R_{B2}	R_{E2}
5 V	2 kΩ	10 kΩ	470 Ω	18 kΩ	1 kΩ	1.5 kΩ

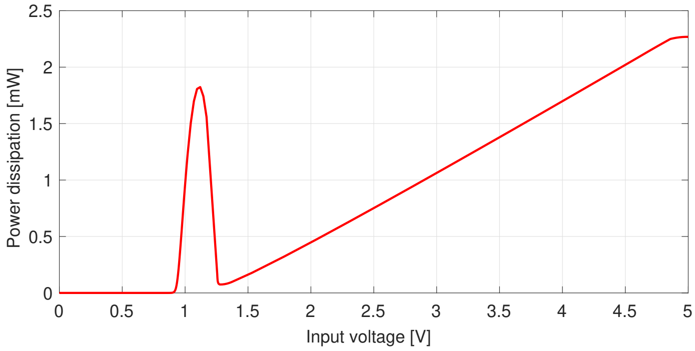


Fig. 4. Switching circuit using two transistors. Time-domain response to linearly increasing input voltage.

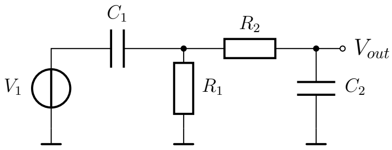
The simulation aims to highlight the particular behavior of the PNP transistor labeled Q_2. The quantity tested is the power dissipation of Q_2, depending on the linear sweep of the input voltage V_1 over time. According to the diagram, an initial peak in the power dissipation curve is experienced as a characteristic feature. The specialty of the circuit is that the dissipation shows a very nonlinear nature, and transistors have different operation modes during the input sweep. It is possible to test how the characteristic operation regions can be detected and how the worst-case values can be found on this circuit.

The next example represents a second-order band-pass filter shown in Fig. 5.

The resulting transfer function is displayed on a Bode plot. In this case, the high-pass and low-pass bands could be identified automatically, and at specific parameter settings, the high-pass and low-pass stages are highly independent; they do not influence each other considerably.

The following example demonstrates an amplitude-stabilized Wien-bridge oscillator circuit shown in Fig. 6.

The example also illustrates the framework's post-processing capability. The upper graphs show the output signal, with red circles indicating the zero crossing points determined by interpolation. The specialty of this circuit is that the operational amplifier's slew rate limit has effect only at particular parameter constellation so that it can be tested, e.g. how the distortion is influenced in certain parameter subspaces,



V_1	C_1	R_1	R_2	C_2
1 V	1 µF	1 kΩ	100 kΩ	100 pF

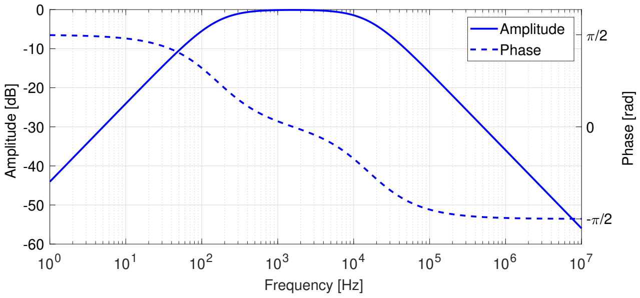
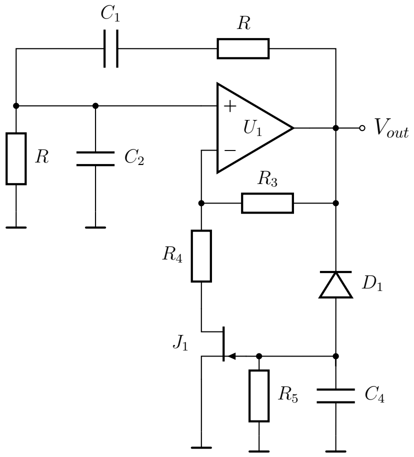


Fig. 5. Band pass filter and its transfer function



R	C_1	C_2	R_3	R_4	J_1	D_1	R_5	C_4	U_1 SR
10 kΩ	100 nF	100 nF	10 kΩ	4.7 kΩ	2N3819	1N4148	10 kΩ	100 nF	0.26 V/µs

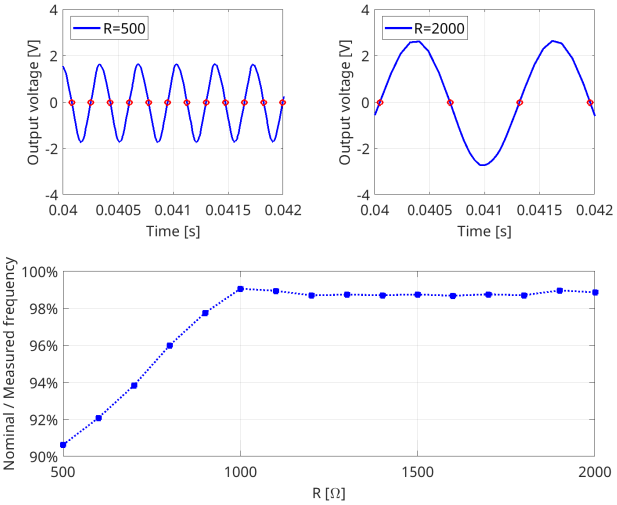
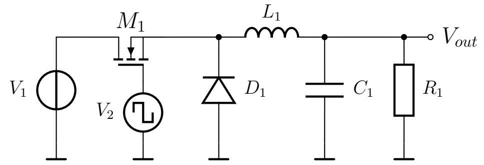


Fig. 6. Wien-bridge oscillator

and the output frequency sensitivity for different parameters is also influenced by the fact whether the slew rate limit is achieved or not. The bottom plot shows the frequency limiting effect of the slew rate for different values of resistor R.

Figure 7 shows a simplified model of a switching mode power supply. It is a buck converter intended to output a lower DC voltage than the input.



V_1	V_2	M_1	D_1	L_1	C_1	R_1
5 V	10 V	BSP89	1N4148	1 mH	1 µH	100 Ω

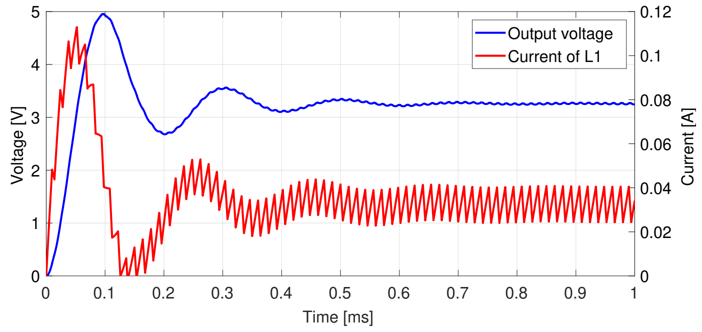


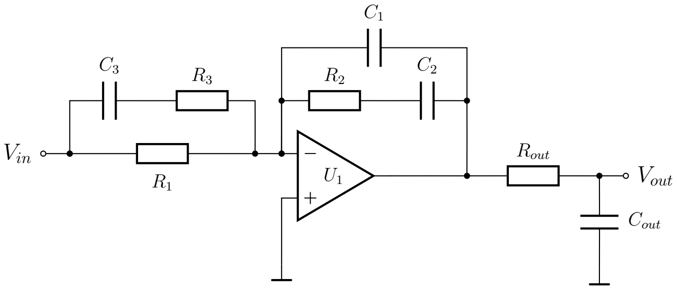
Fig. 7. Simple buck converter and its power-on transient simulation

The simulation represents the power-on transient of this buck converter. The overshoot in both the output voltage and the inductor current can be observed after power-on as well as some oscillations with the voltage and current ripple that persists in a steady state. This circuit allows to test how to distinguish between the high-frequency oscillations and the normal settling of the envelope, the overshoot in the initial transient and the steady state. Generally, the electronic components' parasitic parameters can also affect the resulting signal shapes so that they can be tested during worst-case analysis.

Figure 8 serves as an example of a separable circuit.

Breaking the system into smaller subcircuits reduces the complexity of parameter space, so the worst-case value is easier to find. The example is a linear system; thus, the circuit equations will yield a transfer function in the s-domain, which can be expressed as the fraction of two polynomials. The circuit parameters can be partitioned into disjoint subsets affecting the poles and zeros in the transfer function. Finally, the resulting subcircuits can be evaluated separately based on the parameter groups. There are similar circuits where the overall computation time can be significantly reduced thanks to independent parameters [3].

Figure 9 shows a current limiter with fold-back characteristics. Its primary purpose is to reduce the short circuit current while allowing full output current during regular operation. Its main characteristics are the maximum current and voltage, the slope, and the region near the folding point. In this particular case, the sensitivity to the component parameters proved to be particularly important.



R_1	R_2	R_3	R_{out}	C_1	C_2	C_3	C_{out}
10 kΩ	30 kΩ	560	10 kΩ	6p pF	1.1 nF	3.3 nF	6 pF

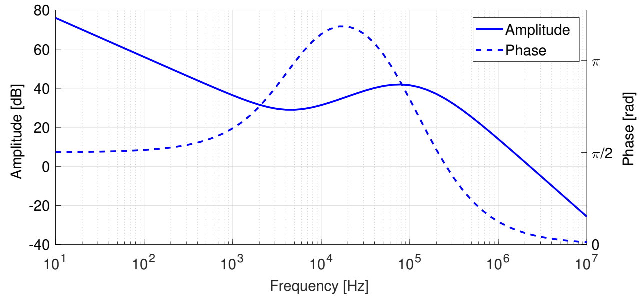
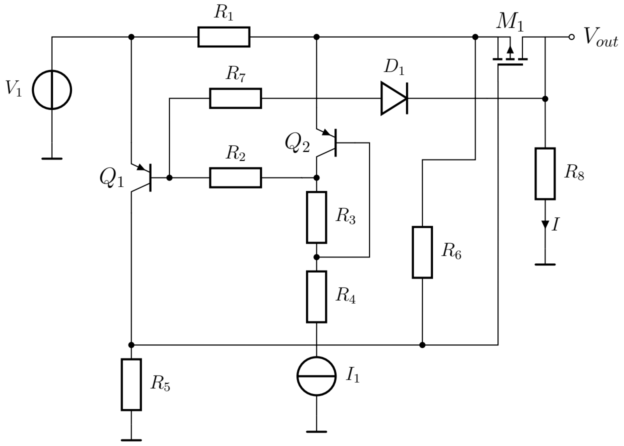


Fig. 8. Active filter circuit for testing separability



R_1	R_2	R_3	R_4	R_5	R_6	R_7
0.25 Ω	2.2 kΩ	400 Ω	2.4 kΩ	47 kΩ	470 kΩ	100 kΩ
R_8	Q_1	Q_2	V_6	I_7	D_8	M_1
1...201 Ω	MMBT2907A	MMBT2907A	5 V	0.43 mA	1N4148	IRF4905

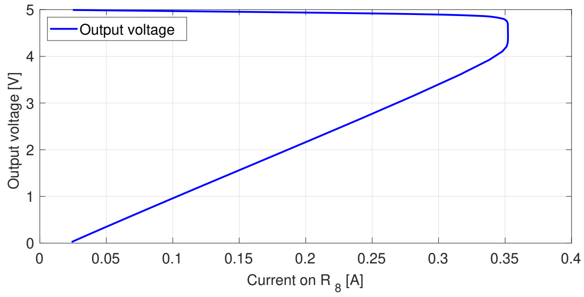


Fig. 9. Current limiter with fold-back characteristics

TABLE I
EXAMPLES OF CIRCUITS WITH TYPICAL BEHAVIOR

Example circuit	**Special property**		
	Nontrivial extreme value	***Separability***	***Characteristic features***
Transistor switch	★		★
Band-pass filter		★	★
Simple buck converter	★		★
Wien-bridge oscillator			★
Active filter	★	★	
Current limiter	★		★

Table I summarizes the above discussion. It shows, how the circuits can be categorized according to the research goals.

In addition to the examples described above, other circuits of varying complexity are also included in the database. Even for structurally simple circuits, the number of parameters increases significantly if complex component models are used. Although it is indeed important to mention how the circuits' complexity affects the software environment's computational requirements, this aspect is not discussed in the paper, as further research is needed to obtain quantitative results. The research goals include the possibility of automatically decomposing complex circuits to some extent, thus reducing the computational complexity.

V. Conclusions

This paper presented a framework designed to improve Worst-Case Analysis processes for electronic circuits. The framework integrates a software environment with a circuit database, addressing challenges in circuit analysis by facilitating automation, optimizing simulation workflows, and supporting a range of analysis methodologies.

The framework provides tools to assist in developing and validating WCA algorithms. The individual software components in the framework are integrated into a coherent system that supports diverse methodologies and advanced analytical approaches, enhancing analysis efficiency and flexibility while runtime optimization is also considered.

The main scientific contribution of this paper is a set of circuits. This database is intended to be used as a test collection to develop advanced analysis methods that can automatically identify typical properties of circuits and apply them in the analysis process. An important consideration in the design of the circuits was the choice of parameters such that the circuits exhibit the desired properties.

Potential extensions of the framework include incorporating machine learning techniques to enhance predictive capabilities and feature extraction. The presented collection of circuits can also be further extended. However, even in this form, they can also serve as a reference and benchmark for testing many analysis methods, as this paper illustrates with examples. Additional features like advanced visualization tools could further support analysis and design tasks.

Overall, the framework provides a structured approach for advancing WCA methods and fosters innovation in automated circuit analysis.

References

[1] B. Johanson, D. Russell, W. Swavely, Reliability Analysis Center, "Worst case circuit analysis application guidelines," Reliability Analysis Center, 1993

[2] B. A. Lenertz, "Electrical design worst-case circuit analysis: guidelines and draft standard (REV A)," AEROSPACE REPORT NO. TOR-2013-00297, The Aerospace Corporation, June 3, 2013.

[3] B. Bak, P. Weisz, Gy. Orosz, "Worst-Case Circuit Analysis using Schematic Conversion and Decomposition," *2024 25th International Carpathian Control Conference (ICCC)*, Krynica Zdrój, Poland, 22-24 May 2024, doi: 10.1109/ICCC62069.2024.10569978

[4] C. M. Rocco, "Variability analysis of electronic systems: classical and interval methods," in *Annual Reliability and Maintainability Symposium*, Philadelphia, PA, USA, 1997, pp. 188-193, doi: 10.1109/RAMS.1997.571704.

[5] N. Femia and G. Spagnuolo, "True worst-case circuit tolerance analysis using genetic algorithms and affine arithmetic," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 9, pp. 1285-1296, Sept. 2000, doi: 10.1109/81.883323.

[6] T. Ding, L. Zhang, R. Trinchero, I. S. Stievano and F. G. Canavero, "Worst-Case analysis of electrical and electronic equipment via affine arithmetic," in *2017 International Conference on Electromagnetics in Advanced Applications (ICEAA)*, Verona, 2017, pp. 991-993,

[7] M. W. Tian and R. C.-J. Shi, "Worst case tolerance analysis of linear analog circuits using sensitivity bands," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 8, pp. 1138-1145, Aug. 2000

[8] M. Sunun and S. Uatrongjit, "Improvement of sensitivity band technique for worst case tolerance analysis of linear circuits," in *2008 5th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, Krabi, Thailand, 2008, pp. 713-716

[9] E. Niculescu, D-M. Purcaru, M. Niculescu, "An approach to worst-case circuit analysis," *WSEAS Transactions on Electronics*, ISSN: 1109-9445, vol. 4, no. 10, Oct. 2007., pp. 237-244

[10] K. Horváth, B. Bank, Gy. Orosz, "Towards Automated Worst-Case Analysis of Circuits: Selecting Initial Values for Global Optimization," *30th Minisymposium, Budapest, Hungary, 2023, 4 p.*

[11] P. Weisz, "Circuit Database for Testing Worst-Case Circuit Analysis Methods" mit.bme.hu https://home.mit.bme.hu/~weisz/circuit-database (accessed Jan. 23, 2025).