INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2025, VOL. 71, NO. 3, PP. 1–6
Manuscript received February 12, 2025; revised June, 2025. doi: 10.24425/ijet.2025.153636

Efficient transient simulation of efuse

Zbigniew Jaworski

***Abstract*—This paper presents a novel approach to model efuse device. An efuse is a simple semiconductor device which can be referred to as a programmable resistor and is available in many modern CMOS technologies. The efuse resistance can be changed by burning procedure, i.e. applying specific electric current value for particular time. The new efuse resistance is retained permanently. The typical efuse application is one-time programmable (OTP) memory. However, process design kits do not provide any model covering the transition from initial to burned state. Thus, verification of programming of an OTP cell is practically impossible. To address this problem, a behavioral Verilog-A model of efuse has been developed. This paper presents the model and its application to verify the example OTP cell designed in 22 nm FD-SOI technology. The proposed model is easy to use and to allow for effective transient simulation of efuse-based designs.**

***Keywords*—Efuse; Verilog-A; transient simulation; OTP memory; FD-SOI; CMOS**

I. Introduction

SEVERAL modern technologies provide designers with an efuse – a simple semiconductor device that can be treated as a field-programmable resistor. The efuse resistance may be changed only once by employing so called burning procedure. The burning is realized by means of applying a specific, relatively high, electric current value for a particular time. The resulting efuse resistance is retained permanently.

The typical field of application of the efuse is design of one-time programmable (OTP) memory. The classic OTP usage is to store digital vector which controls e.g. active mismatch compensation circuitry of an operational amplifier or comparator [1], [2], [3]. However, the only available description of an efuse device is a subcircuit, which can be statically configured to reflect either the initial intact resistance or the resistance of the blown efuse. Usually, process design kits (PDK) usually do not provide a model covering the initial to blown state transition. Thus, verification of the proper operation of an OTP cell in the programming phase is practically impossible. In order to overcome this difficulty, designers use various tricks and *ad hoc* solutions specific to a particular CAD tool. This situation calls for a new systematic approach that holistically addresses the problem.

This paper presents a novel approach to the efuse modeling which eliminates disadvantages of current solutions. The proposition is inspired by the many examples of successful implementation of analog hardware description language (typically Verilog-A) to develop compact models of semiconductor devices, not only transistors but also polysilicon resistors, memristors, sensors, ReRAM etc. [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24]. The usage of Verilog-A(MS) models for analog and mixed-signal systems modeling is currently a well establish approach. All the major CAD system vendors provide analog simulators that fully support Verilog-A model. Moreover, the performance of simulation of Verilog-A models improves systematically. Thus, the idea to choose the AHDL approach to build a compact efuse model seems very rational.

In the following sections the developed Verilog-A model of efuse is presented. The complete code is shown and explained in detail. Next, the example OTP cell designed in 22 nm FD-SOI technology, is demonstrated the practical proposed efuse model performs in real-life case.

This work has been supported by the OCEAN12 (Opportunit to Carry European Autonomous driviNg further with FD-SOI technology up to 12 nm node).

Z. Jaworski is with Warsaw University of Technology, Institute of Microelectronics and Optoelectronics, Poland (e-mail: zbigniew.jaworski@pw.edu.pl).

II. Proposed Solution

The attempt to build a compact model of an efuse device can be only as successful as complete and precise is the information provided by the manufacturer. Unfortunately, process design kits offer very basic subcircuits which represents electrical properties of efuse in the two static states of the device: intact and blown. Moreover, neither information on the actual physics of the burning process is provided, nor are experimental data available. Usually, the documentation gives details on:

- resistance of intact device,
- resistance of efuse after completing burning procedure,
- required programming current value,
- required programming time,
- maximum save sensing current value,
- maximum save sensing time.

A. Assumptions

Since technology providers do not disclose enough data to develop full-featured compact model of efuse as it happens in the case of other solid state devices like e.g. memristors [7], [8], [9], [25]. Thus, a somehow limited but realistic goal has to be defined. In the course of the work, is has been decided that prospected efuse model should:

- achieve full electrical compliance with the subcircuit provided by the PDK, in respect to intact and blown resistance, both in case of DC and TRAN analysis,

- in case TRAN analysis produce reasonable transition from initial to blown state,
- watch for excess burning time,
- watch for excess burning electric current value.

B. The proposed Verilog-A Model of Efuse

The Listing 1 presents the complete Verilog-A code of the *Efuse_AHDL* module. The efuse is basically non-linear resistor. Thus, the model consists of single branch which modeled as following contribution statement which implements the *Ohm's Law* (line no. 78):

```
V(p,n) <+ I(p,n)*Reff;
```

The effective resistance R_{eff} is calculated by the following equation (line no. 75):

```
Reff = Rinit*(1-efuse_state_s) + Rblown*efuse_state_s;
```

The value of *efuse_state_s* variable is obtained by applying the Verilog-A built-in *transient()* function to the *efuse_state* variable to make the transition smoother (line no. 72):

```
efuse_state_s = transition(efuse_state, deltime, transtime);
```

The parameters that control delay and transition time (*deltime*, *transtime*) of the ramp signal are implemented as module parameter (lines no. 28 and 29) to allow user to easily tune the model to actual efuse features. The *efuse_state* variable in the above statement plays the key role in modeling of the burning process. The *efuse_state* is the *integer* type variable and is initiated to 0 value at the simulation time 0 s (line no. 37). The *efuse_state* value will be set to value of 1 when the burning process is completed. This condition is fulfilled when burning process last long enough which means the value of *I_dt* variable rises above *time_th* limit (line no. 65):

```
@above((I_dt - time_th)) efuse_state = 1.0;
```

The value of *I_dt* variable is calculated by means of Verilog-A built-in function *idt(x, initial_condition, clear)* which returns the time integral of its first argument (line no. 62):

```
I_dt = idt(burning & (!efuse\_state), 0, clear);
```

The burning starts when the flow *I(p,n)* excides predefined level, here denoted as *I_th* (line no. 43):

```
@cross( abs(I(p,n)) - I_th, +1, timetol, sigtol)
```

When above event takes place the *burning* variable is set 1 (line no. 48) and the *clear* variable is set to 0 (line no. 47). Since the *efuse_state* was initiated as 0, the *idt()* function starts calculating the time integral of the constant "1" what is equivalent to measuring time of burning process.

When the *I(p,n)* current drops below *I_th* level before the burning is completed, the variables *burning* and *clear* are being reset to initial values 0 and 1, respectively (line no. 52-58). This in turn causes the *idt(x, initial_condition, clear)* function to return to initial condition, in this case 0. The burning procedure is canceled, awaiting a new $I\_dt > I\_th$ event.

In addition, the model watches for excess burning current value (line no. 46) and excess burning time (line no. 68).

Listing 1. Proposed Verilog-A model of efuse device

```
1  //Standard headers are loaded
2  `include "constants.vams"
3  `include "disciplines.vams"
4
5  //Module and ports declarations
6  module Efuse_AHDL (p, n);
7  inout p,n;
8  electrical p,n;
9
10 //Declarations of internal variables
11 integer burning = 0, efuse_state = 0, clear = 0;
12 real I_dt = 0.0;
13 real efuse_state_s = 0.0;
14 real Reff = 0.0;
15
16 //Declarations of module parameters
17 parameter real Rinit        = 125   from [1:inf);
18 parameter real Rblown       = 5000  from [1:inf);
19 parameter real I_th         = 8e-3  from [1e-6:inf);
20 parameter real time_th      = 10e-6 from [1e-6:inf);
21 parameter real time_th2     = 20e-6 from [1e-6:inf);
22
23 parameter real I_th_limit   = 5e-6  from [1e-6:inf);
24 parameter real I_sens_limit = 1e-6  from [1e-6:inf);
25
26 parameter real timetol      = 1e-12 from [1e-12:inf);
27 parameter real sigtol       = 1e-6  from [1e-6:inf);
28 parameter real transtime    = 1e-6  from [1e-6:inf);
29 parameter real deltime      = 0     from [0:inf);
30
31 //Reset module variables the initial simulation step
32 analog begin
33 @ (initial_step or initial_step("static") )
34 begin
35   burning = 0;
36   clear = 0;
37   efuse_state = 0;
38   I_dt=0.0;
39   efuse_state_s = 0.0;
40 end
41
42 //Wait for the event that triggers burning process
43 @ cross( abs(I(p,n)) - I_th, +1, timetol, sigtol)
44 begin
45   $discontinuity(0);
46   if (I(p,n) > I_th_limit) $strobe("Programming current too high");
47   clear = 0;
48   burning = 1;
49 end
50
51 //Stop burning process when current drops prematurely
52 @ cross( abs(I(p,n)) - I_th, -1, timetol, sigtol)
53 begin
54   $discontinuity(0);
55
56   if (burning && !efuse_state) clear = 1;
57   if (!efuse_state) burning = 0;
58 end
59
60 //The idt operator is used to calculate time integral of constant burning=1
61 //The result is equal to burning time
62   I_dt = idt(burning&(!efuse_state), 0, clear);
63
64 //Check if burning time reached required limit and then set efuse_state flag to 1
65 @ above((I_dt - time_th)) efuse_state = 1.0;
66
67 //Warn of excess burning time
68 @ above( (idt(burning, 0) > time_th2 ))
69   $strobe ("Programming mode lasts too long");
70
71 //Control the transition from initial to blown state
72   efuse_state_s = transition(efuse_state, deltime, transtime);
73
74 //Effective resistance of efuse
75   Reff = Rinit*(1 - efuse_state_s) + Rblown*efuse_state_s;
76
77 //Change the efuse resistance from Rinit to Rblown
78  V(p,n) <+ I(p,n)*Reff;
79 end //analog
80 endmodule
```

C. Verification of Proposed Efuse Model

The Figure 1 shows the test bench used to perform example transient simulation of the proposed efuse model. The instance of *Efuse_AHDL* is connected to ideal voltage source and resistor, which emulates internal source resistant, connected in series. The simulation results are presented in Figure 2. The plots in red present test bench level voltages and current. The blue waveforms show internal variables of the *Efuse_AHDL* module.

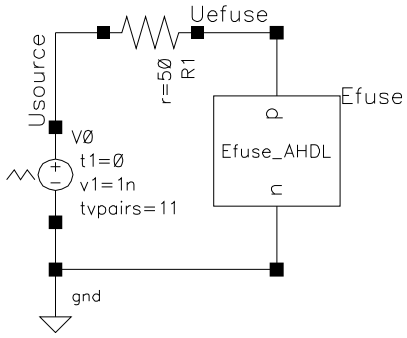

Fig. 1. The test bench schematic for transient simulation of the Efuse_AHDL.

The test scenario consists of two burning attempts. The first one is going to stop before burning is completed while the second attempt will cause the efuse to permanently change its resistance.

In the period $0 \ldots 6\,\mu s$ the value of current *I(p,n)* flowing though the efuse remains below *I_prog* limit and *burning* flag also stays equal to 0. Next, at the time of $6\,\mu s$ the value of the external voltage *Usource* applied to efuse raises and consequently the *Uefuse* goes up to the level that makes the *I(p,n)* current cross the limit and burning procedure starts. The *I_dt* variable, time integral of constant 1, starts to grow linearly. But in the time of $10.5\,\mu s$ the *Usource* drops to $0\,V$ too early. The *clear* signal is set 1 what in turn reset the *I_dt*. The burning is canceled.

The second attempt to burn efuse starts in $14\,\mu s$. This time burning is completed successfully. In the time about $24\,\mu s$ the *I_dt* variable reaches predefined level and the value of *efuse_state_s* variable goes up from 0 to 1. This in turn causes the effective resistance R_{eff} change from R_{init} to R_{blown} level. The change of efuse resistance can be observed both on the *Uefuse* and *I(p,n)* plots.

D. Dealing with discontinuities

The typical problem of Verilog-A models are discontinuities generated by abruptly changing device behavior which in turn causes convergence problems. In the case of efuse model the inevitable discontinuities are created by assigning new values to *efuse_state* variable. The Verilog-A language provides solutions to this problem. Firstly, the *$discontinuity()* built-in function can be used to announce this problem to simulator. Secondly, models should employ filter functions like *transition()*, *slew()* or *laplace_nd()* to smooth discontinuous behavior.

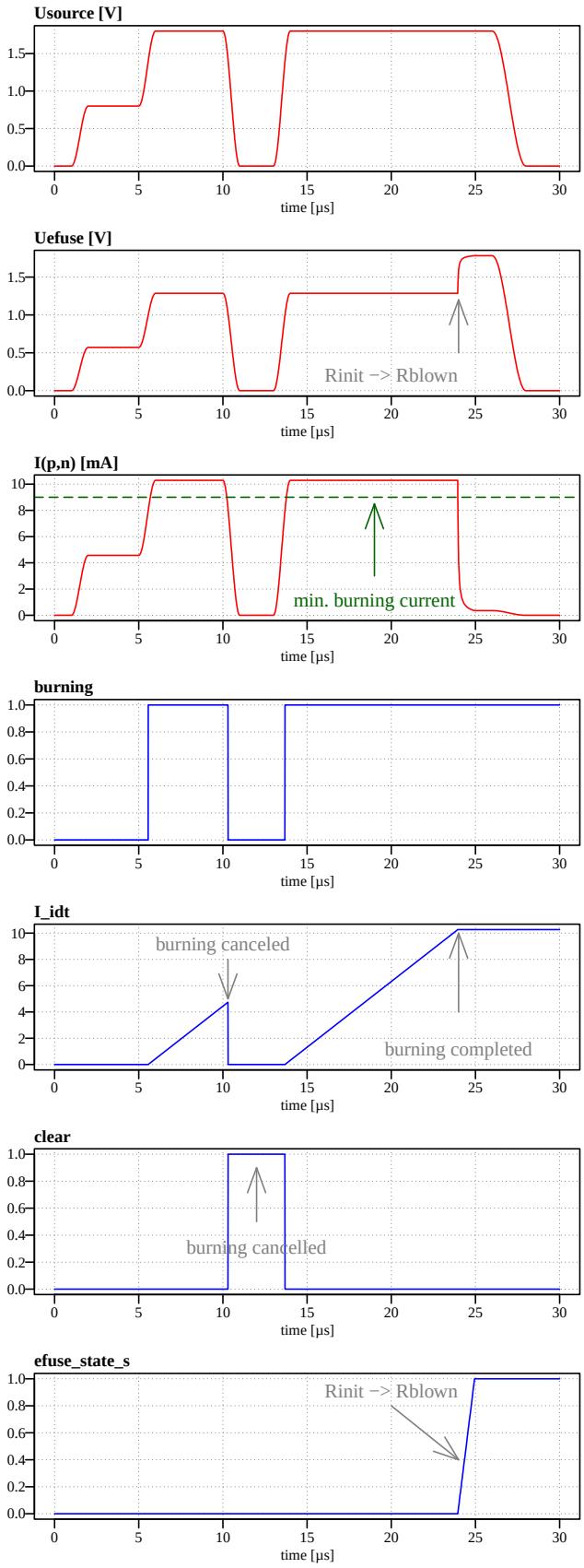

Fig. 2. The results of transient simulation of the Efuse_AHDL test bench.

The presented in the previous subsection efuse model uses the *transition()* function to smooth the transition of *efuse_state* variable and consequently the transition of effective resistance R_{eff}. The Figure 3 shows the example simulation results that illustrates the effects. The R_{eff} sustains continuity but the 1st order differential of R_{eff} does not. The obvious disadvantage of this solution can be clearly visible in the Figure 2. In the time of 24 µs the rapid start of R_{eff} transition is the reason for discontinuity of 1st order of *Uefuse* and *I(p,n)*. This may pose serious convergence challenge or at least may require extremely short time step. Thus, a more

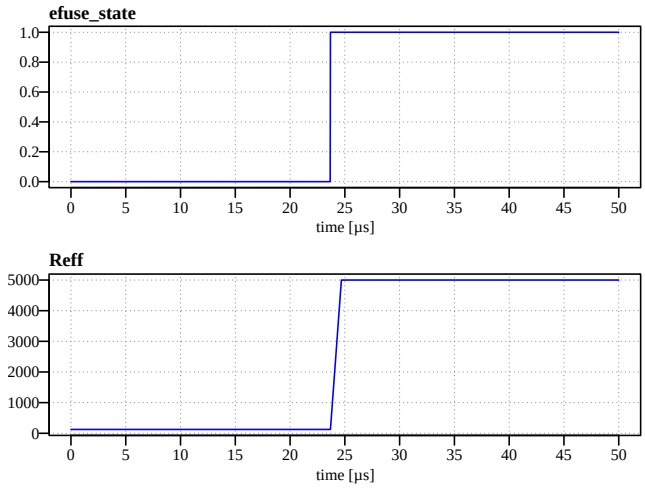

Fig. 3. The effect of *transient()* function used for R_{eff} transition smoothing.

effective smoothing method should be used here. It seems the *laplace_nd()* function is a promising solution. It allows to implement a low-pass filter of arbitrary transfer function defined in the s-domain. An interesting idea is the Bessel filter. The only difficulty here is the calculation of coefficients of the Bessel polynomials [26]. The Table I presents example coefficients of the Bessel denominator polynomial obtained for cut-off frequency $\omega_0 = 1e6$.

TABLE I
BESSEL FILTER DENOMINATOR POLYNOMIAL FOR $\omega_0 = 1e6$

Order	s^0	s^1	s^2	s^3	s^4	s^5
1	1	1*e-6	-	-	-	-
2	1	1.73*e-6	1*e-12	-	-	-
3	1	2.47*e-6	2.43*e-12	1*e-18	-	-
4	1	3.21*e-6	4.36*e-12	3.12*e-18	1e-24	-
5	1	3.92*e-6	6.87*e-12	6.78*e-18	3.81e-24	1e-30

The following statement presents Verilog-A implementation of Bessel low-pass 2nd order filter used to smooth transition of *efuse_state* variable:

```
efuse_state_s=laplace_nd(efuse_state,{1},{1,1.73e-6,1e-12})
```

which substitutes originally used:

```
efuse_state_s = transition(efuse_state, deltime, transtime)
```

The example results of transient simulation of the efuse model utilizing Bessel low-pass filter are shown in Figure 6. The

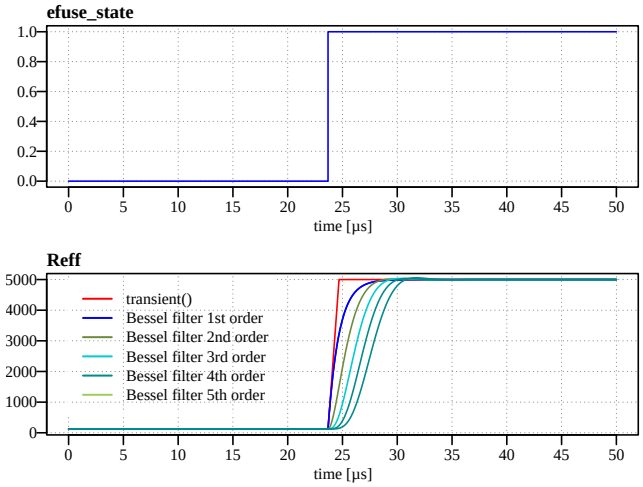

Fig. 4. The effect of employing *laplace_nd()* function to implement Bessel low-pass filter for R_{eff} transition smoothing.

combined plots shown there let us compare various approaches to obtain smooth transition of the effective resistance R_{eff}. It can be clearly observed that Bessel low-pass filter of order 2...5 offers best efuse model performance. There is always a question of the cost of the more sophisticated smoothing.

In order to compare the costs of transient simulation of different smoothing methods a series of experiments has been performed on the workstation equipped with Intel i9-9900K 3.60 GHz CPU and 64GB of RAM and running Centos 7 OS. The experiments were organized as follows. The efuse Verilog-A model was modified to employ one of the six smoothing method: *transistion()* function and five cases of *laplace_nd()* function implementing the Bessel low-pass filter with cut-off frequency $\omega_0 = 1e6$ of orders 1...5. The simulated design was the test bench shown in Figure 1. Each case was simulated five times. The averaged values of total CPU time and peak memory usage for each case are presented in Table II. The results show that memory usage is practically identical in all the cases. However, there is a noticeable but still relatively small increase in the total CPU time. The conclusion is that the smoothing based on usage of Bessel low-pass filter offers much better performance without significant increase of simulation cost.

TABLE II
AVERAGED SIMULATION COST VS SMOOTHING METHOD

Smoothing method	Total CPU time [ms]	Peak memory [MB]
transition() function	9.65	154
Bessel filter 1st order	10.48	155
Bessel filter 2nd order	10.36	154
Bessel filter 3rd order	11.03	151
Bessel filter 4th order	12.12	153
Bessel filter 5th order	12.20	152

III. Example Application of Proposed Model

As it was already mentioned, the typical field of application of an efuse is the design of one-time programmable (OTP) memories. Although efuse based OTP ROMs cannot offer large capacity and would consume unacceptable large silicon area per bit they are still an attractive solution. They do not require complex programming in compare to FLASH-type memories and can be used by an average designer.

One of the popular usage of efuse based OTP is to store relatively short vector of bits that control an analog block, e.g. active mismatch compensation circuitry [1], [2], [3]. The final validation of the developed model should be performed in real-life environment and the memory cell seems to be the obvious choice.

The Figure 5 shows the design of very simple OTP cell realized in 22 nm FD-SOI process, using 1.2 V voltage supply. The instance of the *Efuse_AHDL* module substitutes in this design the actual efuse cell. This OTP cell has two inputs: *prog* and *sense*. The first one controls the burning process while the second switch the OTP cell between power down and read-out modes. The burning procedure requires to activate transistor N0 what happens when *prog=0*. The read-out mode on the other hand requires to set *sens=1*. These two signals should not be active simultaneously. The cell has two outputs: *L* and *R* for demonstration purposes. In practice, monitoring of the *L* output is enough to make effective use of the cell.

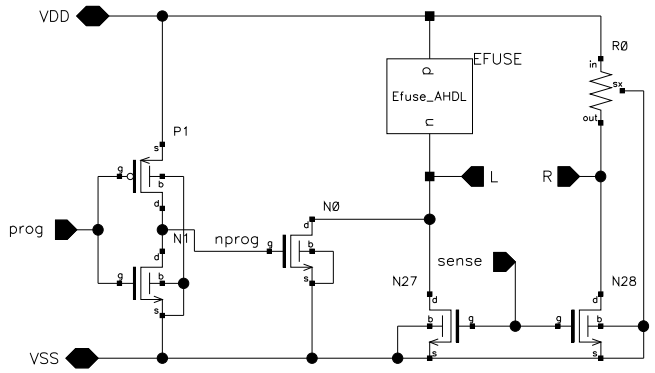

Fig. 5. The simple OTP cell implemented in 22 nm FD-SOI process.

The results of transient simulation of the example OTP cell are shown in Figure 6. The simulation starts with efuse remaining in its initial, intact state. The first attempt to read out the bit stored in the cell takes place around 8 µs and the second one is issued about 58 µs. In the meantime, burning procedure is initiated twice. In addition to OTP cell signal the *burning* and *I_idt* waveforms are presented (plotted in blue) to help to understand what is happening inside the $Efuse\_AHDL$ module. The first burning try (around 25 µs) is too short and efuse returns to initial state. The second attempt to burn efuse (around 40 µs) is completed successfully. It can be clearly observed that first and second read-out operations yield two different voltage values on the output *L*, which can be easily interpreted as logical 1 and 0.

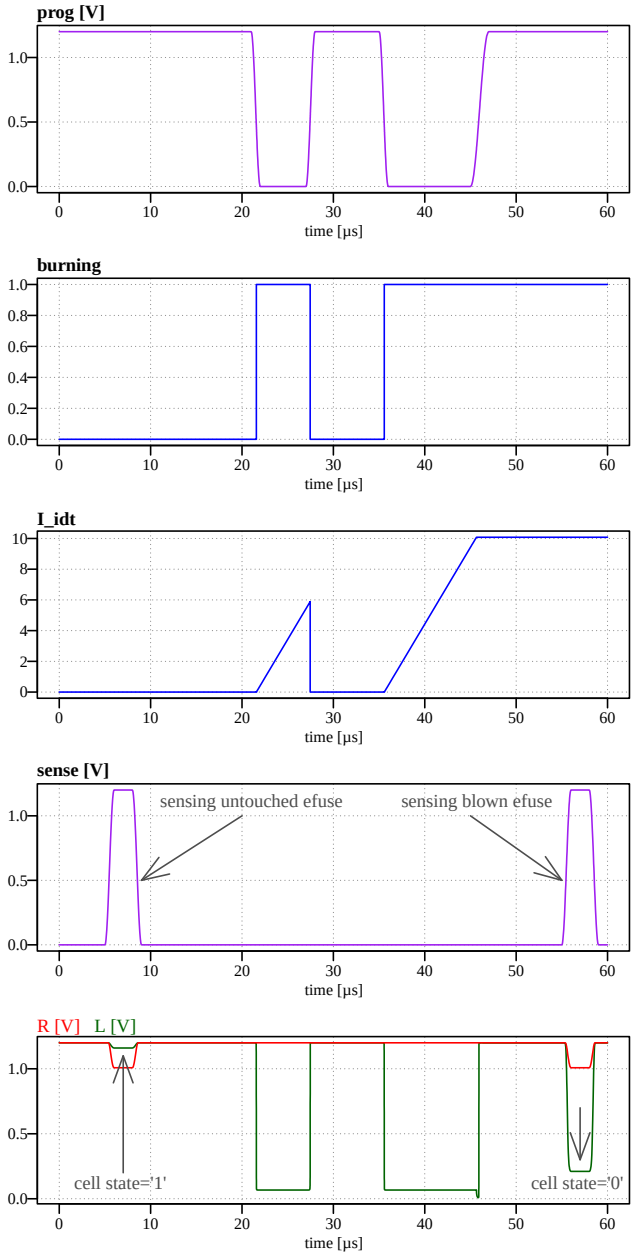

Fig. 6. The results of transient simulation of the example OTP cell.

IV. Conclusions

The presented Verilog-A based efuse model allows to efficiently simulate real-life designs, like OTP memory cell. Since the model is not a part of any PDK technology library potential users need to replace original efuse devices with a custom cell that contains the Verilog-A code. The model is fully parametrized what allows to easily apply it to devices available in various technologies.

The developed model has been successfully tested with several circuit simulators provided by different CAD vendors, with various setup option that affect precision and with different time step control algorithm.

Improvements of the presented efuse model are possible on condition more information on the device physics is made

available to designers. The improvements could include modeling of the parasitic capacitances or the efuse behavior in case of sequence of incomplete burning procedures.

ACKNOWLEDGMENT

This work has been supported by the OCEAN12 (Opportunit to Carry European Autonomous driviNg further with FD-SOI technology up to 12 nm node), a Horizon 2020-ECSEL project no. 783127 jointly founded by the European Commission and Polish National Centre for Research and Development (NCBiR).

REFERENCES

[1] Z. Jaworski, "High resolution latched comparator implemented in 22 nm fd-soi process," in *2018 25th International Conference "Mixed Design of Integrated Circuits and System" (MIXDES)*, 2018, pp. 149–153. [Online]. Available: https://doi.org/10.23919/MIXDES.2018.8436803

[2] ——, "Highly linear 4-bit flash adc implemented in 22 nm fd-soi process," in *2019 MIXDES - 26th International Conference "Mixed Design of Integrated Circuits and Systems"*, 2019, pp. 221–226. [Online]. Available: https://doi.org/10.23919/MIXDES.2019.8787023

[3] Z. Jaworski and P. Wysokinski, "Robustness of digital approach to mismatch compensation in analog circuits realized in nanometer technologies," in *Electron Technology Conference 2013*, P. Szczepanski, R. Kisiel, and R. S. Romaniuk, Eds., vol. 8902, International Society for Optics and Photonics. SPIE, 2013, p. 89020Z. [Online]. Available: https://doi.org/10.1117/12.2031697

[4] O. Z. Kenneth S. Kundert, *The Designer's Guide to Verilog-AMS*, ser. The Designer's Guide Book Series. Springer New York, NY, 2004. [Online]. Available: https://doi.org/10.1007/b117108

[5] D. S. Wladyslaw Grabinski, Bart Nauwelaers, Ed., *Transistor Level Modeling for Analog/RF IC Design*. Springer Dordrecht, 2010. [Online]. Available: https://doi.org/10.1007/1-4020-4556-5

[6] I. M. Dan Fitz Patrick, Ed., *Analog Behavioral Modeling with the Verilog-A Language*. Springer New York, NY, 2007. [Online]. Available: https://doi.org/10.1007/b101821

[7] G. Coram, "How to (and how not to) write a compact model in verilog-a," in *Proceedings of the 2004 IEEE International Behavioral Modeling and Simulation Conference, 2004. BMAS 2004.*, 2004, pp. 97–106. [Online]. Available: https://doi.org/10.1109/BMAS.2004.1393990

[8] H. J. B. da Costa, F. de Assis Brito Filho, and P. I. de Araujo do Nascimento, "Memristor behavioural modeling and simulations using verilog-ams," in *2012 IEEE 3rd Latin American Symposium on Circuits and Systems (LASCAS)*, 2012, pp. 1–4. [Online]. Available: https://doi.org/10.1109/LASCAS.2012.6180334

[9] F. O. Rziga, K. Mbarek, S. Ghedira, and K. Besbes, "An efficient Verilog-A memristor model implementation: simulation and application," *Journal of Computational Electronics*, vol. 18, no. 3, pp. 1055–1064, Sep. 2019. [Online]. Available: https://doi.org/10.1007/s10825-019-01357-9

[10] Z. Jiang, S. Yu, Y. Wu, J. H. Engel, X. Guan, and H.-S. P. Wong, "Verilog-a compact model for oxide-based resistive random access memory (rram)," in *2014 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, 2014, pp. 41–44. [Online]. Available: https://doi.org/10.1109/SISPAD.2014.6931558

[11] M. H. Fino, "Verilog-a compact model of integrated tapered spiral inductors," in *2016 MIXDES - 23rd International Conference Mixed Design of Integrated Circuits and Systems*, 2016, pp. 58–61. [Online]. Available: https://doi.org/10.1109/MIXDES.2016.7529700

[12] B. Wang, C. Li, C.-H. Chen, K. Yu, M. Fiorentino, R. G. Beausoleil, and S. Palermo, "A compact verilog-a model of silicon carrier-injection ring modulators for optical interconnect transceiver circuit design," *Journal of Lightwave Technology*, vol. 34, no. 12, pp. 2996–3005, 2016. [Online]. Available: https://doi.org/10.1109/JLT.2015.2505239

[13] M. Brinson, "Foss compact model prototyping with verilog-a equation-defined devices (vaedd)," in *2019 MIXDES - 26th International Conference "Mixed Design of Integrated Circuits and Systems"*, 2019, pp. 92–97. [Online]. Available: https://doi.org/10.23919/MIXDES.2019.8787063

[14] A. Biswas, D. Ludwig, and M. Cotorogea, "A new computer-aided calibration technique of physics based igbt and power-diode compact models with verilog-a implementation," in *2019 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, 2019, pp. 1–4. [Online]. Available: https://doi.org/10.1109/SISPAD.2019.8870499

[15] M. Reyboz, S. Onkaraiah, G. Palma, E. Vianello, and L. Perniola, "Compact model of a cbram cell in verilog-a," in *2012 12th Annual Non-Volatile Memory Technology Symposium Proceedings*, 2012, pp. 94–97. [Online]. Available: https://doi.org/10.1109/NVMTS.2013.6632872

[16] H. E. Dawale, L. Sibeud, S. Regord, G. Jourdan, S. Hentz, and F. Badets, "Compact modeling and behavioral simulation of an optomechanical sensor in verilog a," *IEEE Transactions on Electron Devices*, vol. 67, no. 11, pp. 4677–4681, 2020. [Online]. Available: https://doi.org/10.1109/TED.2020.3024477

[17] S. R. Nandakumar and B. Rajendran, "Verilog-a compact model for a novel cu/sio2/w quantum memristor," in *2016 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, 2016, pp. 169–172. [Online]. Available: https://doi.org/10.1109/SISPAD.2016.7605174

[18] J. Aguirre-Morales, S. Frégonèse, C. Mukherjee, C. Maneux, T. Zimmer, W. Wei, and H. Happy, "Physics-based electrical compact model for monolayer graphene fets," in *2016 46th European Solid-State Device Research Conference (ESSDERC)*, 2016, pp. 240–243. [Online]. Available: https://doi.org/10.1109/ESSDERC.2016.7599630

[19] S. Zhang, X. Fan, N. Chen, K. X. Wang, J. Xu, and M. Tan, "A verilog-a compact model for four-wave mixing supporting electronic-photonic co-simulation," in *2024 2nd International Symposium of Electronics Design Automation (ISEDA)*, 2024, pp. 16–19. [Online]. Available: https://doi.org/10.1109/ISEDA62518.2024.10617775

[20] A. Bazigos, C. L. Ayala, M. Fernandez-Bolaños, Y. Pu, D. Grogg, C. Hagleitner, S. Rana, T. T. Qin, D. Pamunuwa, and A. M. Ionescu, "Analytical compact model in verilog-a for electrostatically actuated ohmic switches," *IEEE Transactions on Electron Devices*, vol. 61, no. 6, pp. 2186–2194, 2014. [Online]. Available: https://doi.org/10.1109/TED.2014.2318199

[21] O. Prakash, S. Maheshwaram, M. Sharma, A. Bulusu, A. K. Saxena, and S. K. Manhas, "A unified verilog-a compact model for lateral si nanowire (nw) fet incorporating parasitics for circuit simulation," in *2016 20th International Symposium on VLSI Design and Test (VDAT)*, 2016, pp. 1–6. [Online]. Available: https://doi.org/10.1109/ISVDAT.2016.8064852

[22] O. Moldovan, F. Lime, and B. Iñiguez, "A complete verilog-a gate-all-around junctionless mosfet model," in *2015 Conference on Design of Circuits and Integrated Systems (DCIS)*, 2015, pp. 1–4. [Online]. Available: https://doi.org/10.1109/DCIS.2015.7388562

[23] M. A. Fraquet and G. Rafael-Valdivia, "Nonlinear current source model for a gaas transistor implemented in verilog-a using pulsed measurements," in *2020 IEEE MTT-S Latin America Microwave Conference (LAMC 2020)*, 2021, pp. 1–3. [Online]. Available: https://doi.org/10.1109/LAMC50424.2021.9601865

[24] J.-D. Aguirre-Morales, S. Frégonèse, C. Mukherjee, C. Maneux, and T. Zimmer, "An accurate physics-based compact model for dual-gate bilayer graphene fets," *IEEE Transactions on Electron Devices*, vol. 62, no. 12, pp. 4333–4339, 2015. [Online]. Available: https://doi.org/10.1109/TED.2015.2487243

[25] C. C. McAndrew, G. J. Coram, K. K. Gullapalli, J. R. Jones, L. W. Nagel, A. S. Roy, J. Roychowdhury, A. J. Scholten, G. D. J. Smit, X. Wang, and S. Yoshitomi, "Best practices for compact modeling in verilog-a," *IEEE Journal of the Electron Devices Society*, vol. 3, no. 5, pp. 383–396, 2015. [Online]. Available: https://doi.org/10.1109/JEDS.2015.2455342

[26] E. Grosswald, Ed., *Bessel Polynomials*. Springer-Verlag Berlin, 1978. [Online]. Available: https://doi.org/10.1007/BFb0063135