# *Verification and Test of Real Circuits*

## *Structure-Preserving Modelling based on Signal Flow Graph*

Mohamed Denguir, Sebastian M. Sattler
Friedrich-Alexander-University Erlangen Nuremberg (FAU)
Chair of Reliable Circuits and Systems (LZS)
Erlangen, Germany
{mohamed.denguir, sebastian.sattler}@fau.de

***Abstract*—For the verification and functional safety of real systems, e.g. of analog and digital circuits, the function associated with the real structure must be modelled with structural integrity ensured. This means that the consistency of the formally derived and modelled function with the function generated by the real structure must be ensured. In addition, the modelled function must exhibit a behavior that is equal to the function to be realized. In this article, a structure-preserving verification method is at first presented, and illustrated with a real digital circuit by verifying the circuit and respectively testing it for known, self-injected faults. The results are displayed as signal flow graphs by means of self-written program codes.**

***Keywords—Verification; Test; Structure-preserving modelling***

## I. INTRODUCTION AND MOTIVATION

A test program is used to implement a given test specification on a special test target system. The reliability of the test operation itself is ensured by the robust and reproducible execution of the test program. It is necessary to test all paths of the successive parts of a test program in a solid way [1]. However, the testing of technically real structures is becoming more challenging with increasing complexity of the system and its functional diversity. The exact assignment of virtual (test) functionality and real structure should be carried out [2]. It is therefore useful to use a robust and effective verification method that can be applied to test the functionality of complex and targeted real systems. This method is called the "Structure-Preserving Modelling based on Signal Flow Graph".

## II. STRUCTURE-PRESERVING MODELING

The modelling of real systems by the use of functions (using the description of the behavior) that preserve the structure, leads to a presentation in a signal flow graph (SFG). A SFG is the presentation of an abstract algebra, the formulas that can be used to verify structure-preserved images. The SFG is the directed graphical presentation of a multiple set (multi-set), which consists of so-called edges and nodes, which represent morphisms (e.g. functions) and objects (e.g. sets). The phase lists of a SFG consisting of nodes and edges are independent from one another and without restriction of the generality concurrent to each other (simultaneously in parallel) [2]. This data model in positive logic (PL) with the properties associativity (asso) and identity (id) exhibits the prerequisites for encoding a control circuit [3]. This data model splits into Operational (OP) and Control (CTRL) and guarantees specification (spec), test (test), functioning (k) and non-functioning (/k) for each partial structure.

### *A. Steps to create the data model*

The creation of the data model is divided into three steps: First (1), the real pins that occur in the real structure must each be designated with a positive or negative literal, which corresponds to the embedding of the real structure into a coding universe. Then (2), the real structure is abstracted in an event-based manner and in a model simulated by a directional signal flow using two paths (dual-rail). Subsequently (3), the state transitions of the sub-models are described in propositional logic expressions (AA) and their signal flow graphs (SFG) are encoded with (1, 0, -) in three-valued logic (so-called ternary vector lists – TVL [4]).

### *B. Rules for generating the model*

The first step towards the creation of the data model, the labelling of the pins, can be channelled with the "directed" laws of the respective physics or electrical engineering by means of two rules. First and most important rule: naming pins according to pin partitioning [2], these are states (S), primary inputs (PI) and primary outputs (PO). The second rule is to fine-tune the pins according to the transmitted digital signal value (1 or 0), negative literal for signal value equals 0 (low) and positive literal for signal value equals 1 (high). The negative literal is marked with the prefixed symbol "/". Regarding the second step for creating the data model, the representation of the "substitutable" complementarity by means of the operation switch (¬) is carried out in dual rail [5] (see III.C). For carrying out the third step for creating the data model, the coding of the operations spec, test, k and /k in TVL: k and /k require realities that are e.g. components and lines, spec defined limits and test fault models to be known. As each state is splitted into two states called a present state $S^a$ and a next state $S^n$, $S = (S^a, S^n)$ is generally determined in that way, that the operations spec and test are directed from $S^a$ to $S^n$, while the controls k and /k are directed from $S^n$ to $S^a$. Thus, as soon as

978-1-5386-1813-4/17/$31.00 ©2017 IEEE

the spec operation (OP) is executed, a pin or a state retains its value in terms of a state stabilization [6], $S^n = S^a$. By contrast, test is used as operation (OP) (called Fail), the state changes into the same state with a negative literal, $S^n = /S^a$. Since k and /k as controls (CTRL) depend on their respective reality, it is inconvenient to develop a general formula for their implementation. They are modelled explicitly or implicitly (see Section III.D).

### C. Common example

The particular data model is generated from the following design pattern. Let X and Y be two real pins, that are present in a structure, then its associated SFG is shown (in PL) in Fig. 1. The SFG shown in Fig. 1 can be reduced to the SFG in Fig. 2, X = ($X^a$, $X^n$) and Y = ($Y^a$, $Y^n$). TABLE I shows the state transitions as phase lists. The state transitions (events) from defined limits, reality and known errors in TABLE I are encoded in TVL. The star symbol "*" represents "undefined" regarding the entire article. Is a list place reserved with "∗", it means that this place does not exist.

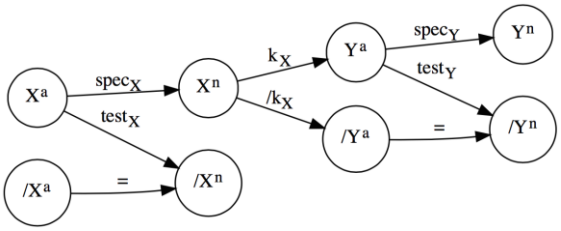



Fig. 1. Common data model (in PL)

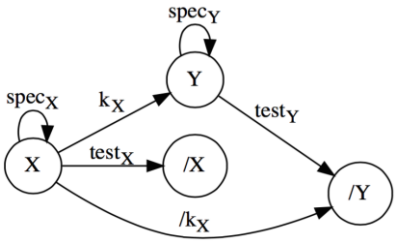



Fig. 2. Reduced data model (in PL)

TABLE I. STATE TRANSITIONS AND FUNCTIONS (IN TVL)

| $S^a$ | | Event | | | $S^n$ | | Function | |
|---|---|---|---|---|---|---|---|---|
| X | Y | defined limits | real world | known faults | X | Y | OP | CTRL |
| 1 | * | (1,0,-) | * | * | 1 | * | $spec_X$ | * |
| * | 1 | * | (1,0,-) | * | 1 | * | * | $k_X$ |
| 1 | * | * | * | (1,0,-) | 0 | * | $test_X$ | * |
| * | 0 | * | (1,0,-) | * | 1 | * | * | $/k_X$ |
| * | 1 | (1,0,-) | * | * | * | 1 | $spec_Y$ | * |
| * | 1 | * | * | (1,0,-) | * | 0 | $test_Y$ | * |

## III. EXAMPLE DIGITAL CIRCUIT

In this chapter, a real digital circuit (DUT - Device-Under-Test) should be prepared for the verification of known, self-defined errors with the help of the "Structure-Preserving Modelling based on SFG". We consider the digital circuit shown in Fig. 3, which is already available on a real board. The circuit is equipped with jumpers, which serve as well-known faults to be modelled. The goal we pursue is to carry out the verification on the basis of deliberately installed errors by plugging in and removing jumpers, respectively. For this purpose, an automatic test device based on a µ-Controller is developed and programmed. In this article, we will restrict ourselves to the theoretical part, the report on the structure-preserving modeling of a circuit using SFG.

### A. Schematic representation of the DUT

The digital circuit (DUT) in Fig. 3 shows CMOS inverters (INV_1 and INV_2) connected in series and controlled by a npn BJT. The circuit includes six switches S1, S2, S3, S4, S5 and S6 serving as jumpers and can be plugged in (closed) or out (open) manually. In order to avoid any undesirable electrical interruptions or short circuits in normal operation, switches S1, S3 and S6 are closed, while switches S2, S4 and S5 are open. If a digital 1 is set at the input (A_CTRL), transistor T1 becomes conductive and Pin B is pulled to GND (digital 0). If switches S1 and S2 are in normal operation, Pin C takes this digital 0. This is then inverted by the inverter (INV_1) and outputted as digital 1 to Pin D. Pin E accepts the digital 1 from Pin D if the switches S3 and S4 are switched to normal operation. This is then inverted by the inverter (INV_2) and outputted as digital 0 to pin F. Pin G at the output accepts the digital 0 from pin F if the switches S5 and S6 are in normal operation. If, on the other hand, a digital 0 is applied to the input, the transistor T1 is non-conductive and Pin B takes the digital 1 out of the ohmic resistors R2 and R3 >> R2 due to the voltage divider. The value is switched to a digital 0 by the inverter (INV_1) and transferred to Pin E. Resulting of inverter (INV_2) the value is switched again to a digital 0 and propagated to Pin G.

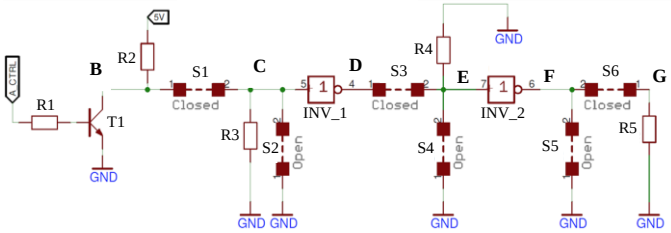



Fig. 3. Schematic of DUT (with courtesy from Liebherr GmbH)

### B. Designate the pins

It is desired to get a digital 1 at the output, but for that a digital 0 must be applied to the input (A_CTRL). Therefore, after taking account of the rules in section II.B, the input pin A_CTRL at the npn-base must be declared as negative literal /A and described as primary input (PI). Consequently, after consideration of the reality (Section III.A), the real pins B, C, F and also G must be declared as positive literals B, C, F and G and the pins D and E as negative literals /D and /E. This results in six pins or states (B, C, /D, /E, F, G) and a primary input (/A). Pin G is both a state and primary output (PO). The switches are declared as positive literal when closed in normal operation, otherwise with negative literal. The list is S1, /S2, S3, /S4, /S5 and S6.

### C. SFG in Dual-Rail

After successfully implementing the first step of the verification method, the designating of the pins, we can now apply the second step, abstracting the reality in event-based manner, and present a SFG in dual-rail using the operation switch (¬). By the operation switch, the four original states (B, C, /D, /E, F, G) become six substitutable complementary states (¬B, ¬C, ¬/D, ¬/E, ¬F, ¬G). On each of the twelve states the function test is applied, which means that twelve substitutable, complementary states can be achieved. In summary, the states B, C, /D, /E, F and G result in /B /C, D, E, /F and /G and from ¬B, ¬C, ¬/D, ¬/E, ¬F and ¬G results /¬B, /¬C, /¬/D, /¬/E, /¬F and /¬G. Fig. 4 shows the SFG. The state (5V, /GND) in the SFG represents the voltage supply. From state (5V, /GND) to state B and dual to that to state ¬B follows by the reality $k_{5V}$ und $k_{/GND}$.

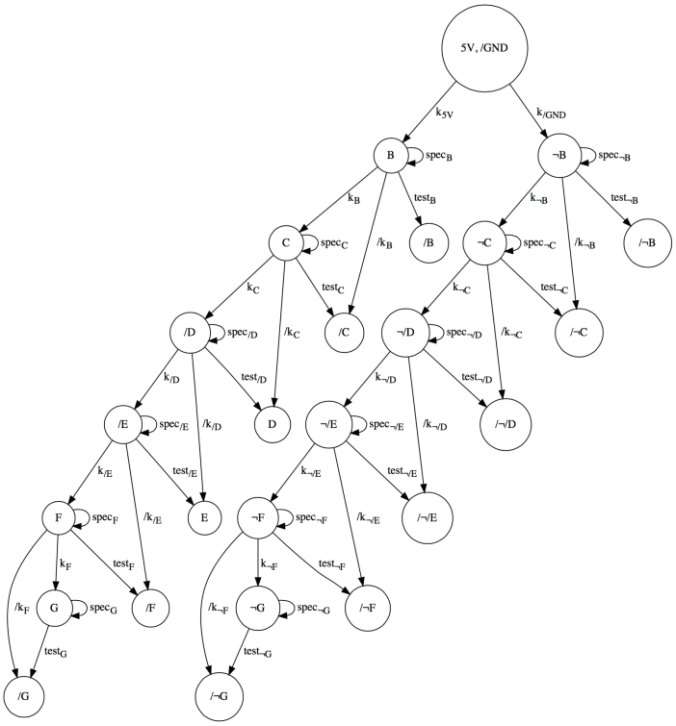



Fig. 4. Data model (SFG) of the DUT

### D. Coding of the functions (OP, CTRL)

The following coding-tables TABLE II to TABLE VII serve together with their SFG as more detailed description of weighted edges of the SFG from Fig. 4. They exhibit the specific encoding of the Operational Functions spec and test, and Control Functions k and /k. This is done, after taking into account the rules from section II.B, analogous to the structure of TABLE I, $\delta(S^a, S^n)$ depends on the states (B, C, /D, /E, F, G), the primary input (/A) and the fault models (S1, /S2, S3, / S4, / S5, S6).

In TABLE II, the state B retains its digital 1 if the function $spec_B$ is fulfilled. This is done without influence of the errors, i.e. (S1, /S2) = [∗ ∗], the value of the primary input /A is set to digital 0. If the function $test_B$ is fulfilled, state B changes to digital 0. This is due to the errors (/A, S1 /S2) = [0 1 0] and (/A, S1 / S2) = [1 - -]. In the case of the first combination (S1, /S2) = [1 0], a known fault is detected as present, switch S2 is closed (electrical short circuit). In the other case, (S1, /S2) = [-1], it could occur (S1, /S2) = [1 1], causing the known fault to be correctly recognized as not present. In this way, the data model contains all the answers to known errors detected as existing faults and known errors detected as non-existent faults.

Since power supply is always assumed to exist, $k_{5V}$ and $k_{/GND}$ are considered as fulfilled and do not require coding (related SFG to TABLE II).

On the other hand $k_B$, $/k_B$ and $k_{¬B}$ are explicitly modelled in TABLE III without consideration of fault models. For $k_B$ or $k_{¬B}$ modelled with (C, B) = [1 1] or (C, B) = [0 0], if for example the component R3 is present and for $/k_B$ with (C, B) = [0 1], for example if R3 << R2.

TABLE II. OP ENCODED IN B

| B | /A | S1 | /S2 | B | Function |
|---|---|---|---|---|---|
| 1 | 0 | ∗ | ∗ | 1 | **$spec_B$** |
| 0 | 1 | ∗ | ∗ | 0 | **$spec_{¬B}$** |
| 1 | 1 | - | - | 0 | **$test_B$** |
| 1 | 0 | 1 | 0 | 0 | |
| 0 | 0 | 1 | 1 | 1 | **$test_{¬B}$** |
| 0 | 0 | 0 | - | 1 | |

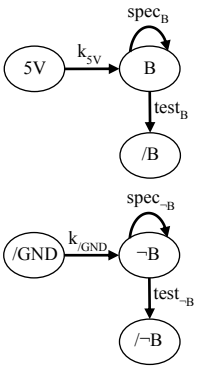



TABLE III. (OP, CTRL) ENCODED IN (C, B)

| C | S1 | /S2 | C | Function |
|---|---|---|---|---|
| 1 | ∗ | ∗ | 1 | **$spec_C$** |
| 0 | ∗ | ∗ | 0 | **$spec_{¬C}$** |
| 1 | - | 0 | 0 | **$test_C$** |
| 1 | 0 | - | 0 | |

| C | S1 | /S2 | B | Function |
|---|---|---|---|---|
| 1 | ∗ | ∗ | 1 | **$k_B$** |
| 0 | ∗ | ∗ | 1 | **$/k_B$** |
| 0 | ∗ | ∗ | 0 | **$k_{¬B}$** |

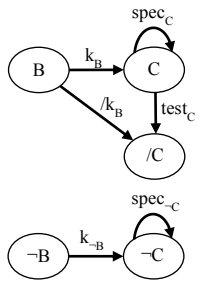
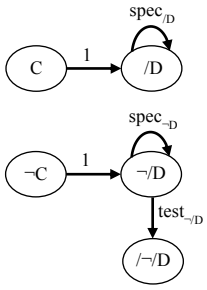



TABLE IV. OP ENCODED IN /D

| /D | S3 | S4 | C | /D | Function |
|---|---|---|---|---|---|
| 0 | ∗ | ∗ | 1 | 0 | **$spec_{/D}$** |
| 1 | ∗ | ∗ | 0 | 1 | **$spec_{¬/D}$** |
| 1 | 1 | 0 | ∗ | 0 | **$test_{¬/D}$** |



The coding of the functions $k_C$ and $k_{\neg C}$ is already covered by $spec_{/D}$ and $spec_{\neg/D}$ with (/D, C, /D) = [0 1 0] and (/D, C, /D) = [1 0 1]. These are the edges from C to /D and ¬C to ¬/ D in the corresponding SFG in 0. However, $k_F$ and $k_{\neg F}$ are explicitly modelled in TABLE VII and are holding the output stage F. For the coding of additional functions similar procedure applied.

TABLE V. (OP, CTRL) ENCODED IN (/E, /D)

| /E | S3 | /S4 | /E | Function |
|---|---|---|---|---|
| 0 | * | * | 0 | **$spec_{/E}$** |
| 1 | * | * | 1 | **$spec_{\neg/E}$** |
| 1 | - | 0 | 0 | **$test_{\neg/E}$** |
| 1 | 0 | - | 0 | |

| /E | S3 | /S4 | /D | Function |
|---|---|---|---|---|
| 0 | * | * | 0 | **$k_{/D}$** |
| 1 | * | * | 0 | **$/k_{/D}$** |
| 1 | * | * | 1 | **$k_{\neg/D}$** |

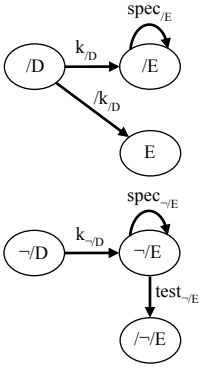



TABLE VI. OP ENCODED IN F

| F | /S5 | S6 | /E | F | Function |
|---|---|---|---|---|---|
| 1 | * | * | 0 | 1 | **$spec_F$** |
| 0 | * | * | 1 | 0 | **$spec_{\neg F}$** |
| 1 | 0 | - | * | 0 | **$test_F$** |

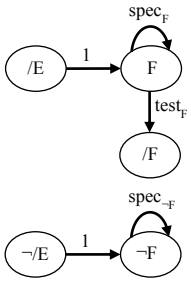
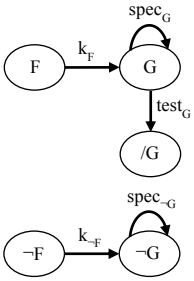



TABLE VII. (OP, CTRL) ENCODED IN (G, F)

| G | /S5 | S6 | G | Function |
|---|---|---|---|---|
| 1 | * | * | 1 | **$spec_G$** |
| 0 | * | * | 0 | **$spec_{\neg G}$** |
| 1 | - | 0 | 0 | **$test_G$** |
| 1 | 0 | - | 0 | |

| F | /S5 | S6 | F | Function |
|---|---|---|---|---|
| 1 | * | * | 1 | **$k_F$** |
| 0 | * | * | 0 | **$k_{\neg F}$** |




Not all functions or state transitions are encoded in the SFG in Fig. 4. ($/k_C$, $/k_E$, $/k_F$, $/k_{\neg B}$, $/k_{\neg C}$, $/k_{\neg/D}$, $/k_{\neg/E}$, $/k_{\neg F}$) do not lead to any added value, ($test_{/D}$, $test_{/E}$, $test_{\neg C}$, $test_{\neg F}$, $test_{\neg G}$) will never be fulfilled. Therefore, the states D, /¬C, /¬F and /¬G are not achieved in Fig. 4. The merging of the corresponding SFG in TABLE II, TABLE III, 0, TABLE V, TABLE VI and TABLE VII gives the SFG in Fig. 5.

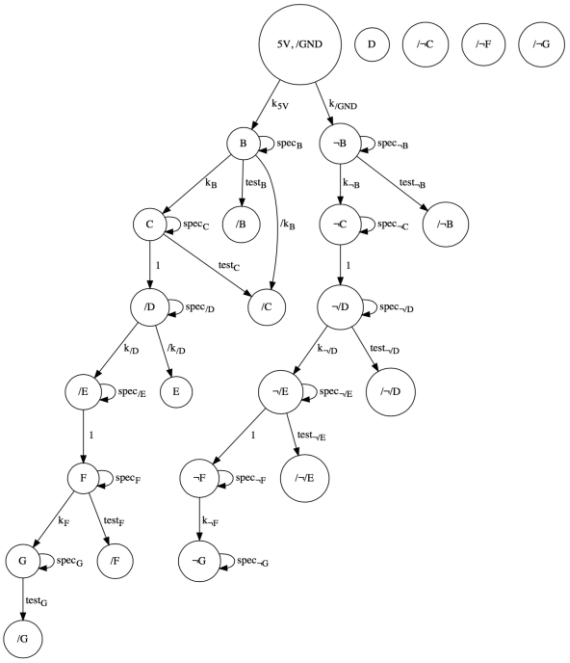



Fig. 5. Data model (SFG) of the DUT

## *E. Use cases*

In this section, the expected results of the step-by-step verification method for the predefined digital circuit are now presented. Through a self-written program in VBA (Visual Basic for Applications) is-values shall be compared to set-values from TABLE II to TABLE VII. The program visualizes the corresponding SFG, coming from the resulting data model from Fig. 5 with the exception of four not reached states D, /¬C, /¬F and /¬G. All phase lists (edges with their nodes) in this SFG are without restriction of the generality as adjoining. The operations spec, test, k and /k are coloured - if they are fulfilled - as green, red, black and blue arrows. If they are not, dashed arrows serve as their visualization. In the first case (Fig. 6) all switches S operate in normal mode (the known faults are not present), so obtaining a digital 1 in the Is-table. Input /A has a digital 0. Since spec is fulfilled regarding B, C, /D, /E, F and G, each state (/B, /C, D, E) maintains its value ($S^n = S^a$) and the corresponding edges are represented as green arrows. Similarly, k regarding /B, /D and F are fulfilled. The remaining functions are not fulfilled due to the different actual- and set values and are marked accordingly. In the second case (Fig. 7) input A is set to digital 1, which in the first step results in the takeover of digital 0 at B. Consequently spec is no longer fulfilled regarding /B, for this test is fulfilled. However, since all switches are in normal operation, known faults are recognized as non-existent faults. In the third case (Fig. 8) /S2 is set to closed, receiving consequently the digital 0 in the first step. The corresponding SFG is identical to the SFG from the second case, whereas now a well-known error is recognized as an existing error. The Is-table for the fourth case (Fig. 9) is filled similar to the first case, so that now spec is fulfilled in the dual rail regarding ¬B, ¬C, ¬/D, ¬/E, ¬B and ¬C.

| $S^a$ | | | | | | PI | Switch | | | | | | $S^n$ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| B | C | /D | /E | F | G | /A | S1 | /S2 | S3 | /S4 | /S5 | S6 | B | C | /D | /E | F | G |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |

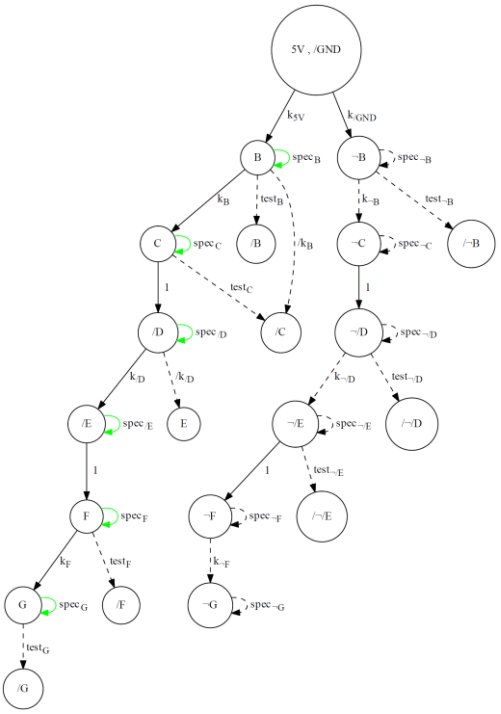



Fig. 6. Case 1: Is-table and associated SFG

| $S^a$ | | | | | | PI | Switch | | | | | | $S^n$ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| B | C | /D | /E | F | G | /A | S1 | /S2 | S3 | /S4 | /S5 | S6 | B | C | /D | /E | F | G |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |

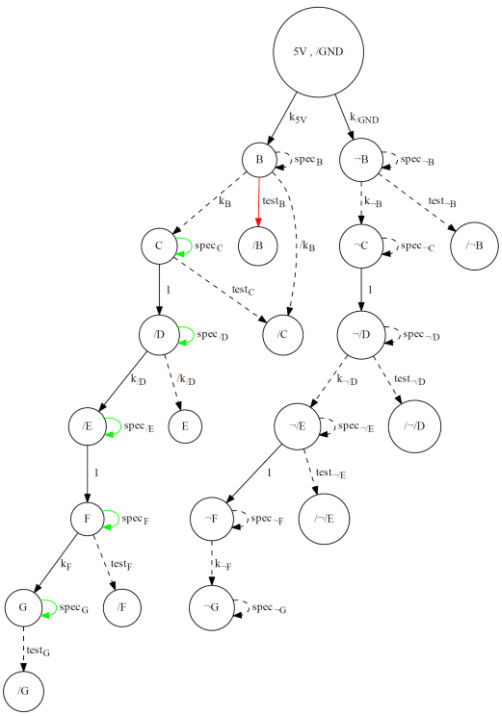



Fig. 7. Case 2: Is-table and associated SFG (1st step)

| $S^a$ | | | | | | PI | Switch | | | | | | $S^n$ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| B | C | /D | /E | F | G | /A | S1 | /S2 | S3 | /S4 | /S5 | S6 | B | C | /D | /E | F | G |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |

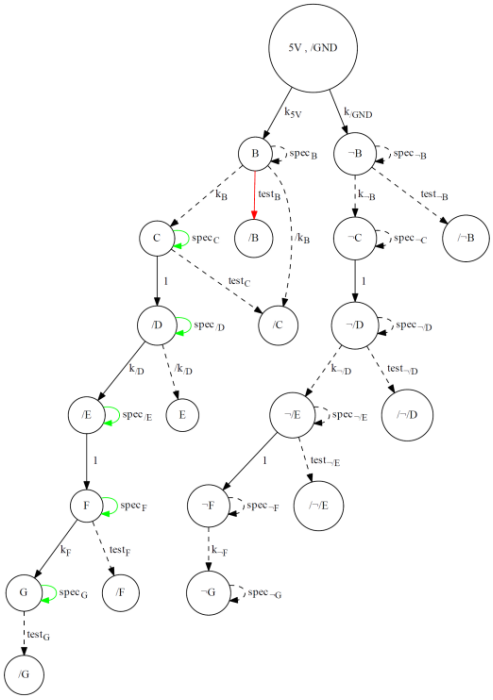



Fig. 8. Case 3: Is-table and associated SFG (1st step)

| $S^a$ | | | | | | PI | Switch | | | | | | $S^n$ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| B | C | /D | /E | F | G | /A | S1 | /S2 | S3 | /S4 | /S5 | S6 | B | C | /D | /E | F | G |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |

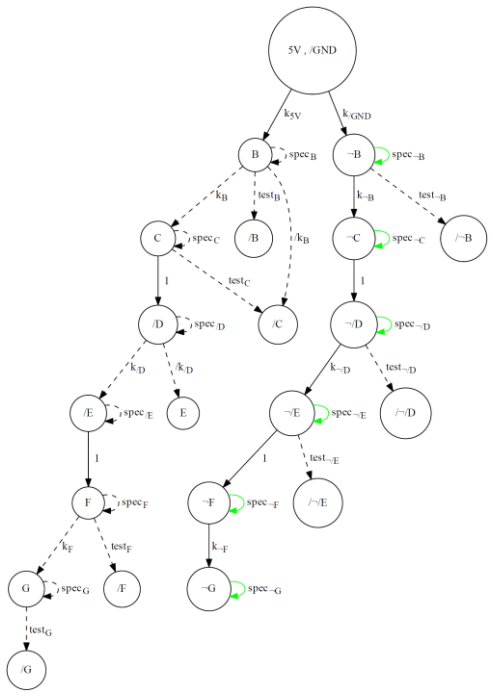



Fig. 9. Case 4: Is-table and associated SFG

In the last case (Fig. 10) /S4 is set to closed, so it receives consequently the digital 0. Hence, spec is no longer fulfilled regarding ¬/D, but the test is fulfilled. In this case, it is still to mention, that /k regarding /D is fulfilled because of the accordance of its is- and set-values and marked accordingly (blue). Similarly, other examples can be generated.

| $S^a$ | | | | | | PI | Switch | | | | | | $S^n$ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **B** | **C** | **/D** | **/E** | **F** | **G** | **/A** | **S1** | **/S2** | **S3** | **/S4** | **/S5** | **S6** | **B** | **C** | **/D** | **/E** | **F** | **G** |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |

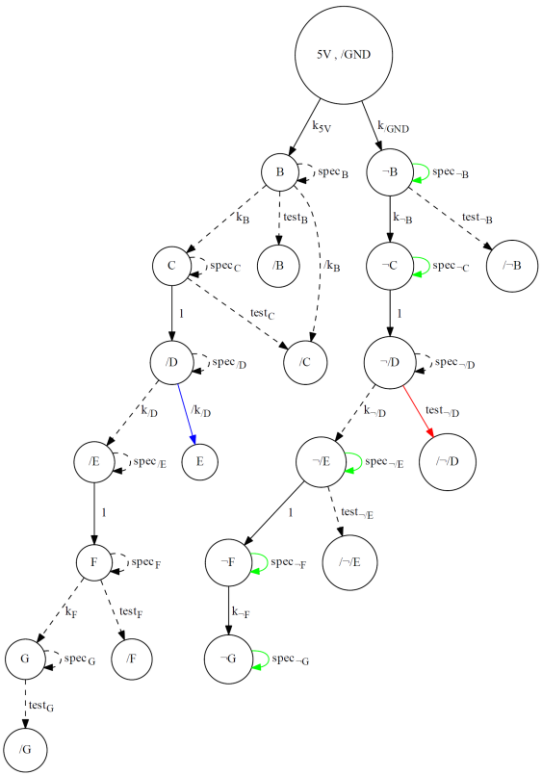



Fig. 10. Case 5: Is-table and associated SFG (1st step)

The examples presented show how to take care of analog properties like voltage and current. It is known as parametric measures. Indeed, the methodology is not limited to those numbers. It can additionally be applied to other physical constraints as well as all kind of event based digital forms. To be mentioned should delay, energy and power. The effort there will also be in the modeling of the underlying structure in an abstraction of a signal flow graph (SFG).

## IV. Summary and Outlook

The Structure-Preserving Modelling based on Signal Flow Graph, which is presented in this paper, is a structure-preserving verification method, which is used to test systems or circuits for known faults. By Structure-Preserving Modelling, we mean the consistency of the formally derived function with the real function actually generated. The verification can be carried out, with the aid of certain derived rules, in three steps: from the embedding of the real model into a coding universe, by specifying a data model as SFG in dual rail, up to the creation of the submodels (OP, CTRL) in AA.

In comparison to known methods like simulation and validation the proposed methodology takes care of the direction of the structure and can therefore preserve the structure of the device under test within its directed structure. In our terminology this is called verification. It has the opportunity to go for algebraic methods that are closed under idempotency, so called lattices. It also enables to write fast and solid code. Whether it outperforms up-to-date approaches in the field is not yet proven.

The TVL created in section III.D can be arranged and summarized as Quaternary vector lists (QVL) in a next step. This happens by replacing symbol "∗" representing "undefined" in TABLE II to TABLE VII by the format symbol "x". Thus QVL is encoded in (1, 0, -, x). It is then possible to program search functions that go through the QVL, determining certain criteria, such as test coverage, defect coverage and test severity. However, the QVL generated by a complex reality can take on great proportions in its dimensions. Therefore, the database (QVL) should be compiled (without information loss) line-by-line [7] in order to ensure lower memory requirements and shorter computing time.

This line-by-line compacted database is for its functional portion multidimensional regarding columns. The columns show the dependencies or correlations between the individual functions (spec, test, k, /k) and their possible fault models. Thus, it is useful to examine such correlations also from the aspect of classification methods, which can classify in dimensions of data sets. This allows the Principal Component Analysis and the Linear Discriminant Analysis [8].

## References


[1] Gurkaynak, F.; Villiger, T.; Oetiker, S.; Felber, N.; Kaes-lin, H.; Fichtner, W.: Functional Test Methodology for Globally-Asynchronous Locally-Synchronous Systems. 8th Symposium on Asyn. Circuits and Systems (ACS). (2002)

[2] Uygur, G.; Sattler, S.: Structure Preserving Modeling for Safety Critical Systems. IEEE 20th International Mixed-Signals Testing Workshop (IMSTW). (2015)

[3] Siegfried, W.: Operationszustand versus Steuerzustand - eine äußerst zweckmäßig Unterscheidung. Technische Universität Kaiserslautern. (2000)

[4] Posthoff, Ch.; Steinbach, B.: Logic Functions and Equations. Springer. (2004)

[5] Gössel, M.: Automatentheorie für Ingenieure. Akademie Verlag. Berlin. (1991)

[6] Zander, H.J.: Logischer Entwurf binärer Systeme. VEB Verlag Technik. Berlin. (1989).

[7] E. McCluskey, Logic Design Principles (Prentice-Hal, Englewood Cliffs, New Jersey) (1986)

[8] Xie, Y.; Zhang, T.: A fault diagnosis approach using SVM with data dimension reduction by PCA and LDA method. Chinese Automation Congress (CAC). pp. 869 - 874. (2015)