Contribution to the Methods and Algorithms of Automatic Generation of Functional Tests for Processors

Ján Hudec*
Institute of Computer Systems and Networks
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Ilkovičova 2, 842 16 Bratislava, Slovakia
jan.hudec@stuba.sk

Abstract

The dissertation thesis is aimed at automatic functional test generation methods for processors. Testing processors needs continually new test generation methods, algorithms and test application techniques for their verification, manufacture testing and reliable life-time run. The functional tests are mainly generated over an instruction set architecture and processor model description. They run in similar form and frequency as real programmes. Such tests are categorized as the software-based tests. A metric for quality evaluation of the software-based tests is obviously done by code coverage of a described processor model. A simulation tool is used with the mentioned metric calculation.

A new test generation method is presented in the dissertation thesis. It uses VHDL model of processors and genetic algorithms with adaptation of various evolutionary strategies. In addition fitness function is defined for evaluation of generated tests by genetic algorithms and a new method for creation of a starting test (the first population) has been proposed. The described methods are suitable for automatic generation of software-based tests for processors. Functionality and effectiveness of the developed methods were evaluated in implemented software system AGenMIX over one type of RISC processors. Two representative experiments are shown in the paper.

Categories and Subject Descriptors

B.7.3 [**Reliability and Testing**]: Test generation

*Recommended by thesis supervisor: Associate Professor Elena Gramatová. Defended at Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava on December 18, 2014.

Hudec, J. Contribution to the Methods and Algorithms of Automatic Generation of Functional Tests for Processors. Information Sciences and Technologies Bulletin of the ACM Slovakia, Vol. 6, No. 4 (2014) 24-30

Keywords

testing, functional test, test generation, software-based testing, genetic algorithm, evolutionary strategy, VHDL, code coverage

1. Introduction

As system on chip (SoC) architectures have become widely accepted for complex digital and hybrid systems, the problem of testing embedded processor cores has opened a crucial problem. Incoming technologies, chip complexity and increasing clock frequencies give new challenges for testing not only the whole SoCs but also the embedded processors integrated there. Functional tests are mainly important for processors during their verification, as additional manufacturing tests and also during their life time testing. Thus increasing design complexity opens research and development of functional test pattern generation (TPG) again as one of the most challenging issues in both areas: verification and testing not only SoCs but mainly processors integrated there. Current industrial practices mostly use random test programs (functional tests). The TPG methods need both manual and guided random tests. Manual generation of functional tests can be a tedious process and random tests do not guarantee specified faults coverage (e.g. functional or stuck_at, delay faults etc.). The need of more deterministic functional or hybrid (deterministic + random) test generation approaches is desirable. However, the deterministic functional TPG is a long–standing open problem. Classical test methods based on sequential and structural automatic test pattern generation (ATPG) consume too computational time for complex digital circuits and thus also for nowadays processors. Thus a progress comes around research and practical cases for developing new functional test generation methods with their automatization.

The TPG methods targeted to software–based and self-testing over instruction set architecture (ISA) have been developed and published. During last decade the TPG methods marked as software-based self–testing (SBST) methods have been developed for processor models [5]. Self–testing means that the tests can be performed on a tested processor without using an automatic test equipment. Such SBST methods produce additional tests to structural tests of processors and for life–time run on SoCs. The tests for SBST consist of various ordered sequences of instructions from ISA of a tested processor and specified data as operands if necessary.

Minimal or even no circuit modifications are needed for running SBST performed in normal operating mode. Moreover once a test code (test programs, tests for SBST) has been uploaded in the tested processor then the test programs are completely autonomous and can run at speed.

A new test generation method for SBST has been developed using different types of genetic algorithms (GAs) with specific first population and changing possibilities of evolutionary strategies (ES). The ES changing is based on feedback quality of generated test programs evaluated by defined fitness function. The major features of the new test generation method are adaptability, flexibility and lower computational time for finding optimal functional tests with accepted functional or stuck–at fault coverage. A metric used for functional fault coverage evaluation in SBST is based on code coverage of a modelled processor. Some rules and strategies have been defined for developing an effective and flexible ATPG system for generation of test programs. The new method is based on language VHDL (Very High Speed Integrated Circuits Description Language) and ISA of processors. A new system, named AGenMIX, has been developed with link to the simulator ModelSim for code coverage calculation [9]. Experiments were done over DP32 RISC type processor [2].

The paper is organized as follows. The next section presents a related work about functional testing processors. The third section describes motivation and goals of the dissertation thesis. Basic definitions, notation and the new developed test generation method are presented in section four and five respectively. Selected two experimental results over the DP32 processor model in VHDL and its ISA are presented in section six. The last section concludes the paper with a summary of theoretical and practical contributions obtained in the dissertation thesis. The list of references and author's publications from the dissertation thesis core are involved at the end of the paper.

2. Functional testing processors - background

There is a long history on functional testing processors and microprocessors. Fault coverage is low in application of the earlier developed TPG methods for functional testing. They have started more from behavioural level and not from register transfer level (RTL) or structural level of processor models. Recently, some research activities have been initialized towards functional testing using mainly RTL. The processor applies functional test programs named test mixes in the paper (this notation is defined and presented in section four) using its native ISA. The test mixes run at the processor's actual operating frequency.

Various methods have been presented for the functional TPG based on different fault models [4], [5], [12], [13], [16]. The latest use genetic algorithms (GAs) with one selected evolutionary strategy (ES) and randomly generated a first population (starting test programs/test mixes) [6], [7], [10], [14], [15]. Fault coverage quality is basically evaluated by code coverage calculation and using a professional simulator or a professional fault simulator based on stuck-at fault models (stuck–at 0, stuck–at 1). The functional ATPGs are mostly evaluated by covering activation of registers during data transfer based on the processor RTL model. It was published that if tests for SBST show high code coverage then also can achieve high coverage of stuck–at 0, stuck–at 1 faults. Experimentally it has been achieved that app. 90 % code coverage is sufficient for such software–based tests [14]. Automatization of the developed SBST methods is essential aspect in testing processors.

One of the first automatized functional test generation method is based on a system graph representing registers (nodes) and data transfers (arcs) and they are evaluated by mnemonic code of instructions [19]. Evaluation of the proposed method was realised over the processor core DP32 (the 32–bit RISC type processor) and by an in–house structural fault simulator based on stuck–at fault coverage (96 % fault coverage has been achieved by the described method) [3]. The other two functional test generation methods use abstraction models of processors like RTL models and their ISA. The methods are based on analysis of data dependencies of available SBST programs and some parameters of the tested processors. These methods are targeted to pipeline processor architectures. Experiments shown remarkable results up to 95 % in stuck-at fault coverage for two RISC processors (MiniMIPS and OpenRISC 1200) [8]. Similar ideas were presented with experiments over the Sun OpenSPARC microprocessor [1].

The next methods for automatic test programs generation use a selected GA and ESs application with feedback information targeted to generation of new populations using simulators [6], [7], [14], [15], [18]. The idea presented in the papers has been transformed into implementation of a new ATPG system, named μGP. The μGP system composes of a test generator at RTL, a fault simulator for stuck-at faults, ISA library and processor division into modules. The evolutionary strategy ES($\mu + \lambda$) was used in the generator where μ is parents population and λ is new individuals for generating a new population, (new test programs/new test mixes). The new individuals are achieved by means of tournament selection for their reproduction [18]. The μGP system was also used for automatic test programs generation for 32–bit processor Leon2 with SPARC V8 architecture [7]. The simulator ModelSim was used for simulating the processor design and calculating the statement coverage. The fitness function was defined as the direct measure of the test program's coverage. Another case study and results with μGP have been described in [6] and [15]. The μGP effectiveness were tested also over the Intel i8051 processor core with 93,6 % structural fault coverage using on the academic fault simulator FENICE [18]. Some results for μGP and the DLX/pII processor core are presented in [14]; the ModelSim simulator [10] was used as the external evaluator with different metrics. The obtained results were shown up to 90 % in statement and branch coverage and above 80 % in condition and toggle coverage.

The SBST characteristics and advantages are [5]:

- **Non-intrusiveness:** SBST does not need any processor modification and no extra power consumption is produced in comparison to the normal operation mode.
- **At-speed testing:** Test program application and responses evaluation are performed at the processorÂt's actual frequency, which enables examination of faults that are not observable at lower testing frequencies.

- **No over-testing:** SBST concentrates on the same circuitry used in the functional processor modes and therefore avoids test overkill, and thus detection of faults that would never manifest during the normal processor operation. It leads to significant yield gains and also shorting test time.
- **In-field testing:** Self-test programs from manufacturing testing can be reused in the field throughout product lifetime which is very important in nowadays complex applications.

The SBST methods show these disadvantages [4]:

- Some faults can modify the test program flow and potentially can lead to an endless execution, making it difficult to take back the control of the system at the end of the test.
- Some memory addresses are never accessible because they are not reserved to the test procedure, therefore resulting in a coverage loss.
- Size and execution time of the test could be prohibitive in case of stringent real-time application requirement.
- IP (intellectual property) protection is not guaranteed, since the test program may reveal details about the processor core implementation.

All SBST and functional test generation methods using GAs generate test programs by using one type of GAs and one applied evolutionary strategy ES($\mu + \lambda$). The first population is generated randomly in all published SBST TPG methods. Obviously the main drawback of these methods is high computing complexity and therefore there is open space for other research.

3. Motivation and objectives

The key idea of the dissertation thesis is development of a new functional test generation method for SBST based on GAs with adaptation of various evolutionary strategies. Besides of basic evolutionary strategies ES($\mu + \lambda$) and ES(μ, λ) it should be useful to use more sophisticated ES defined recently. This evolutionary strategy increases usability of other specified parameters as life-time of population (k) and probability of mutation (p) with the goal to use changing the mentioned strategies.

The latest evolutionary strategy ES(μ, k, λ, p) is defined in [17] and it hasn't been used in practical GAs applications and not in ATPG methods till now. Thus, it was main motivation for new GAs application in functional test generation. The strategy ES(μ, k, λ, p) uses:

- Pseudorandom selection of r individuals ($r \geq \lambda \geq \mu$) from μ parents.
- Executing operations by genetic operators.
- Making an order of the obtained individuals based on fitness selection of λ individuals to the new created population with the best fitness in accordance of selected value of parameter $k(k \geq 1)$ - lifetime of individuals (test mixes) in particular generations.

The ES changing should allow finding maximum faster than using only one basic ES. Thus adaptability, flexibility and optimal test generation for SBST can be improved. The goals of the dissertation thesis were to develop and implement the new SBST generation method suitable for its automatic implementation based on this new ES and to develop a new method for generation of the first population more deterministically than randomly.

4. Basic definitions and notation

Basic terms used in the proposed method for SBST and in GA application are described in the section.

A test mix is a valid sequence of finite and ordered instructions that can be run in normal execution instruction mechanism (similar to functional programs).

A test for processor consists of a set of test mixes.

Test mix length is number of instructions in a test mix.

Term "test mix" is used instead of term "test program" used in published methods because both terms have been defined in the same time [10].

Code coverage is defined as coverage of different types of VHDL statements in processor model description. The basic code coverage types are:

- statement coverage,
- branch coverage,
- condition coverage,
- toggle coverage.

Function fitness for one test mix M_j, is defined by (1) with condition described by (2) for *j = 1, 2, ..., N*, where N is number of test mixes.

$$F_j = w_s.s_j + w_b.b_j + w_c.c_j + w_t.t_j, \tag{1}$$

where w_s, w_c, w_b and w_t are weights of code coverage types (*statement, branch, condition, toggle*) and s_j, b_j, c_j, t_j are relevant values of code coverage by test mix M_j received by simulation. Values of w_s, w_c, w_b and w_t are from <0, 1>, and their sum is 1.

$$w_s + w_c + w_b + w_t = 1 \tag{2}$$

The advanced evolutionary strategy ES(μ, k, λ, p) is used in GA application. Parameters μ, λ are parents and new population of generated test mixes, k is a life-time of individuals (test mixes) and p is probability of mutation. If $k = 1/k = \infty$ in ES(μ, k, λ, p) then ES(μ, λ)/ES($\mu + \lambda$) is predetermined.

Diversity is a measure for characterizing an existed population of test mixes and indicates how many different types of them are involved in a new population.

Evenness is a quantitative measure of type representations in existing population which quantifies how equal is the population of testing mixes numerically.

First population (starting individuals) for running GA in SBST generation is a set of test mixes.

The Shannon-Wiener index [17] and/or standard deviation have been defined for diversity and evenness computation. Both types have to be specified for SBST method in application to a VHDL processor model for receiving an optimal test with sufficient (good) code coverage.

Besides of using the new ES, it is important to find the best first population. It can be generated randomly as it is used in the published SBST method till now. The idea to create this population as best as possible means that GA application can find the best test mixes in a shorter time. Three strategies have been defined in the above context.

Strategy 1. Deterministic or pseudorandom generation of an initial started population (test mixes) with higher code coverage can increase quality of the final functional test consisting several test mixes.

Strategy 2. Application of the advanced ES in which all the basic ESs can be combined using different genetic operators for selection and combinations of crossover and mutation can improve finding optimal final solution for test mixes and thus for functional testing processors.

Strategy 3. The fitness function defined more sophistically in GAs can contribute to better feedbacks in GAs run.

Other important parameters for generating the test for SBST have to be specified. They are:

- Number of test mixes or a code coverage limit for ending their generation based on a specified value of the code coverage.
- ES alternation, used GA operators, a first population.
- Weights and the code coverage specification for fitness calculation.
- Test mix length depending of ISA complexity.
- Dependability of instructions sequence in a test mix.
- Operands selection for instructions, if any.

5. Adaptive ATPG method for processors

The general diagram of a new functional ATPG method is presented in figure 1. Inputs to test generation is VHDL model of a tested processor, parameters for test mixes and for GA application described in previous section, and databases of instructions and operands for instructions involved in test mixes. The blocks "Automatic test mixes generation" and "Test mixes evaluation and ES adaptation" are linked to the simulator ModelSim.

The block "Initial test mixes generation" produces a starting set of test mixes selected by one of the implemented method (*random, priority, grouping*). User can select one of them for a tested processor. A lot of experiments were done using these methods over one type processor (DP32) and the most efficient and suitable method was *grouping*. All results are presented in the dissertation thesis. The main principle of the method *grouping* is following: All instructions of a tested processor ISA are divided into several groups based on their characteristics, e.g. instruction with zero, one or two operands, branch instructions, or based on instruction functionality e.g. arithmetic operations, logic operations etc. Then a test mix is generated by adding one instruction from each group based on a specified test mix length. Each instruction has the same probability of its occurrence. There is possible also to use instruction prioritization.

The main block "Automatic test mixes generation" is a core of the SBST method and is activated by the initial test mixes as the first population.

Then GA with genetic operators (selection/ reproduction, mutation, crossover) is performed on the first population of test mixes, and with reference to the fitness value of each mix calculated by ModelSim. The parents for the next population of test mixes are selected by changing parameters and stated ES. Some constraints such as elitism of individuals with changing parameter k in ES (μ, k, λ, p), tournament and roulette selection are also applicable. If the feedback values in the GA progress indicate the small value of diversity or high value of evenness then alternation of parameters in GA have to be adopted. Thus continual adaptation of input parameters k (life-time of test mix/elitism) and p (probability of mutation) and the selection methods (roulette, tournament etc.) are applied until the higher fitness values of generated test mixes are achieved. This procedure is repeating until the best code coverage of test mixes or the maximal number of populations (generations) is achieved.

There is a lot of craftsmanship in definition and assessment of fitness function and the GA parameters. In the presented SBST method there is combination of various constraints and code coverage for estimating the parameters of the fitness function: evaluation statement coverage, branch coverage, condition coverage and toggle coverage. The evolutionary scheme in used GA is very comprehensive and is based on changing advanced strategy ES (μ, k, λ, p).

6. Implementation and experimental results

The described SBST generation method was implemented and integrated in a new software environment and tool AGenMIX implemented in C# language using Microsoft Visual Studio and Microsoft .NET Framework on the platform Microsoft Windows. All inputs/outputs, except for the evaluating the individuals use the XML format. XML allows usability of standard tools such as browsers, constraints inspection, data library and options for populations and configuration. The VHDL language is used for each tested processor for which also ISA library, database of operands, specification of parameters and ESs, they have to be created. The system AGenMIX is adaptable to various ES and can be extended by other blocks or functionalities, if necessary. A final functional test consists of the best test mixes in optimal length with sufficient code coverage.

DP32 is a RISC type 32-bit processor core was used for all experiments presented in the dissertation thesis. Its ISA contains 20 types of instructions with the length of 32 or 64 bits that are frequently used in programs. It is a typical representative of ARM (Advanced RISC Ma-

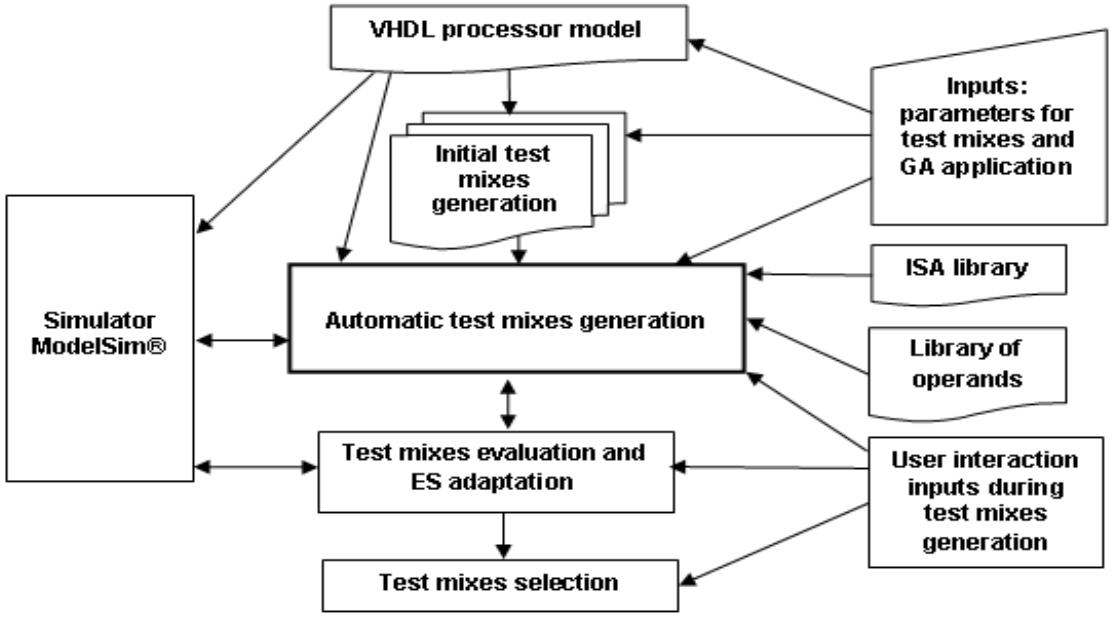

Figure 1: General diagram of automatic SBST generation - AGenMIX

Table 1: Parameters for experiment 1

Parameters	Value
Number of instructions in test mix	20
Number of individuals in a population	12
Number of generation	15
Method–GA	fix
Parameter p	0, 1
Parameter k	∞
Elitism	1
Code coverage metric	statement

Table 2: Parameters for experiment 2

Parameters	Value
Number of instructions in test mix	20
Number of individuals in a population	12
Number of generation	15
Method–GA	feedback
Parameter p	0, 1
Parameter k	3
Elitism	0
Code coverage metric	statement

chine) processor architecture. The processor core is described as a synthesizable VHDL model for academic and research purposes. Specification and architecture of DP32 is open, portable, non-proprietary and scalable to embedded processors, all sharing the same core (non-privileged) instruction set. All experiments use the statement coverage for fitness evaluation. Two selected experiments from the dissertation thesis are described over processor DP32 for visibility of feedback efficiency integrated into the test mixes generation.

Experiment 1. Generation of test mixes for DP32 by GA without feedback from fitness evaluation and changing the basic evolutionary strategy. Input parameters are presented in table 1. The first population (starting test mixes) has been generated by method *grouping*. In this experiment also elitism was used during all generation of new populations.

Maximum statement coverage of the best test mix for specified number of generation is 42.2 %. This low value depends on quality of elite individuals and their impact cannot be blocked off till end of the generation. For higher value it should be increase the limit or to use another ES.

Experiment 2. Generation of test mixes for DP32 by GA with feedback from fitness evaluation and changing the basic evolutionary strategy. Input parameters are presented in table 2. The first population (starting test mixes) has been generated by method grouping. The elitism was constrained by parameter k and its value was specified randomly.

Maximum statement coverage of the best test mix after defined limit for number of generation is 82.9 %. The achieved results show a significant improvement of the statement coverage for the defined generations number (15) using feedback and changing evolutionary strategies. Experiments concerning first population generation presented in the dissertation thesis shown the best quality of the method *grouping* [11]. Complex experiments for DP32 processor and its ISA, described in the dissertation thesis, achieved maximum 95,67 % in the coverage statement.

7. Conclusion

The main contribution of the dissertation thesis is development of the new complex and adaptive functional test generation system based on GAs and the advanced evolutionary strategy defined as ES(μ, k, λ, p). The developed methods, algorithms have been implemented in the software tool AGenMIX for automatic software-based test generation for RISC type processors modelled by VHDL.

The new system is useful for verification and testing processors. The test mixes (test programmes) for proces-

sors can be generated in shorter time than using similar methods with one specified ES, a random generated first population. Functionality and effectiveness of the new implemented TPG method and the system AGenMIX have been tested on the DP32 RISC type processor and its ISA using the professional simulator ModelSim. Experimental results confirmed effectiveness of various ESs combination, more genetic operators for receiving an optimal set of test mixes on qualitative higher level for a tested processor.

The partial contribution is in the new generating first population method (starting test mixes) by the proposed method *grouping*. A lot of experiments with other methods (*random, priority, grouping* with various modifications etc.) shown the best efficiency of the method *grouping*. The final test mixes can be generated in shorter time with higher code coverage.

Theoretical contributions of the dissertation thesis are in application of the advanced ES with their parameters (diversity and evenness) in software-based test generation for processors and variety of first population generation with the goal to receive the final test mixes in shorter time with sufficient high code coverage. The implemented system AGenMIX is flexible to specified parameters, scalable to processor models and their ISA and allows the users generate test mixes with interactions. The system AGenMIX can be used for practical verification purposes at the academic level.

Future works are targeted to the experiments with other type of processor (e.g. RISC processor DLX - 32 bit processor with pipeline instruction architecture).

Acknowledgements. This work has been supported by Slovak National Research Grant Agency under project no. VEGA 1/1008/12 Optimization of low-power design of digital and mixed integrated systems and project no. VEGA 1/0649/09 Security and reliability in distributed computer systems and mobile computer networks.

References

[1] A. Apostolakis, M. Psarakis, D. Gizopoulos, A. Paschalis, and I. Parulkar. Exploiting thread-level parallelism in functional self-testing of cmt processors. In *14th IEEE European Test Symposium ETS 2009*, pages 33–38. IEEE, May 2009.

[2] P. Ashenden. *The VHDL Cookbook*. University of Adelaide, Adelaide, Australia, 1990.

[3] V. V. Belkin and S. G. Sharshunov. Isa based functional test generation with application to self-test of risc processors. In *9th IEEE Workshop on Design & Diagnostics of Electronic Circuits & Systems (DDECS 2006)*, pages 73–74. IEEE, April 2006.

[4] P. Bernardi, L. M. Ciganda, E. Sanchez, and M. S. Reorda. Mihst: a hardware technique for embedded microprocessor functional on/line self-test. *IEEE Transactions on Computers*, PP(99):1–12, 2013.

[5] P. Bernardi, M. Grosso, E.Sanchez, and M. Reorda. Software-based self-test of embedded microprocessors. In *Design and Test Technology for Dependable Systems-on-Chip, Chapter 15*, pages 339–359. Information Science Reference, IGI Global, Herschey, New York, 2011.

[6] P. Bernardi, E. Sanchez, M. Schillaci, G. Squillero, and M. S. Reorda. An evolutionary methodology to enhance processor software-based diagnosis. In *2006 IEEE Congress on Evolutionary Computation*, pages 859–864. IEEE, July 2006.

[7] F. Corno, E. Sanchez, M. S. Reorda, and G. Squillero. Automatic test program generation. *IEEE Design & Test of Computers*, 21(2):102–109, 2004.

[8] D. Gizopoulos, M. Psarakis, M. Hatzimihail, M. Maniakos, A. Paschalis, A. Raghunathan, and S. Ravi. Systematic software-based self-test for pipelined processors. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 16(11):1441–1453, 2008.

[9] M. Graphics. ModelSim SE Command Reference Manual. Mentor Graphics Corp. http://www.supportnet.mentor.com/support/. [Online; accessed September, 2009].

[10] J. Hudec. Vlsi system test design: The methods, problems and experience in microprocessor testing using aftg. In *16th Int. Conference Information Technology Interfaces ITI'94*, pages 191–193. UCC Zagreb, IEEE, June 1994.

[11] J. Hudec. Some results in automatic functional test design for processors. In *Emerging Trends in Computing, Informatics, Systems Sciences, and Engineering, Lecture Notes in Electrical Engineering, Vol. 151*, pages 965–972. New York: Springer Science + Business Media B.V., 2013.

[12] P. S. Kabiri and Z. Navabi. Effective rt/level software/based self/testing of embedded processor cores. In *IEEE 15th International Symposium on Design and Diagnostics of Electronic Circuits & Systems, DDECS 2012*, pages 209–212. IEEE, April 2012.

[13] M. Psarakis, D. Gizopoulos, E. Sanchez, and M. S. Reorda. Microprocessor software-based self-testing. *IEEE Design & Test of Computers*, 27(3):4–19, 2010.

[14] E. Sanchez, M. S. Reorda, and G. Squillero. Efficient techniques for automatic verification-oriented test set optimization. *International Journal of Parallel Programming*, 34(1):93–109, 2006.

[15] E. Sanchez, M. Schillaci, G. Squillero, and M. S. Reorda. An enhanced technique for the automatic generation of effective diagnosis-oriented test programs for processor. In *Design, Automation & Test in Europe Conference & Exhibition 2007, DATE 07*, pages 1–6. IEEE, April 2007.

[16] M. Schölzel, T. Koal, S. Röder, and H. Vierhaus. Towards an automatic generation of diagnosis in field sbst for processor components. In *14th IEEE Latin-American Test Workshop, LATW 2013*, pages 1–6. IEEE, April 2013.

[17] H. P. Schwefel and T. Bäck. Evolution strategies. In *Genetic Algorithms in Engineering and Computer Science*, pages 111–140. John Wiley & Sons Ltd, Chichester, 1995.

[18] G. Squillero. Microgp–an evolutionary assembly program generator. In *Genetic programming and evolvable machines*, pages 247–263. New York: Springer, 2005.

[19] S. M. Thatte and J. A. Abraham. A methodology for functional level testing of microprocessors. In *8th International Symposium on Fault-tolerant Computing*, pages 90–95. IEEE, 1978.

Selected Papers by the Author

J. Hudec. Some Results in Automatic Functional Test Design for Processors. In T. Sobh, K. Elleithy, eds. *Emerging Trends in Computing, Informatics, Systems Sciences, and Engineering, Lecture Notes in Electrical Engineering*, Vol. 151, pages 965–972, ISBN 978-1-4614-3557-0, New York: Springer Science + Business Media B.V. 2013.

J. Hudec. An Efficient Technique for Processor Automatic Functional Test Generation based on Evolutionary Strategies. In V. Luzar-Stiffler, ed. *33rd International Conference on Information Technology Interfaces ITI 2011 Proceedings*, pages 527–532, ISBN 978-953-7138-20-2, IEEE Catalog Number CFP11498-PRT, Cavtat/Dubrovnik, Croatia, June, 2011, UCC Zagreb&IEEE 2011.

J. Hudec. Processor Functional Test Generation - Some Results with using of Genetic Algorithms. In V. Vranic, ed. *2nd Eastern European Regional Conference on the Engineering of Computer Based Systems ECBS-EERC 2011 Proceedings*, pages 159–160, ISBN 978-0-7695-4418-2, IEEE Catalog Number CFP11ECB-CDR, Bratislava, Slovakia, September, 2011, IEEE Computer Press, Los Alamitos: IEEE Computer Society 2011.

J. Hudec. Some results in automatic functional test design for VLSI circuits. In V. Luzar-Stiffler, ed. *32nd International Conference on Information Technology Interfaces ITI 2010 Proceedings*, pages 635–638, ISBN 978-953-7138-18-9, IEEE Catalog Number CFP10498-PRT, Cavtat/Dubrovnik, Croatia, June, 2010, UCC Zagreb&IEEE 2010.

J. Hudec. Functional Test Generator for VLSI Circuits. In A.Popov, R.Romanski, ed. *19th International Conference on Systems for Automation of Engineering and Research SAER-2005 Proceedings*, pages 46–50, ISBN 954-438-501-0,Varna, Bulgaria, Sept. 2005, TU Sofia, EBSCO Publishing 2005.

J. Hudec. Design Methodology and Practice of VLSI Functional Test Synthesis. In D.Kalpic, V.H. Dobric, ed. *23rd International Conf. on Information Technology Interfaces ITI 2001 Proceedings*, pages 461–466, ISBN 953-96769-3-2, Catalog Nr. 01EX491, Pula, Croatia, June 2001, SRCE Zagreb&IEEE 2001.

J. Hudec, M. Hustava. Optimized Design and Implementation of the Toroidal Neural Processor in the Xilinx Virtex FPGA Environment. In A.Popov, R.Romanski, ed. *15th International Conference on Systems for Automation of Engineering and Research SAER-2001 Proceedings*, pages 177–184, ISBN 954-8329-33-6,Varna, Bulgaria, Sept. 2001, TU Sofia, EBSCO Publishing 2001.

J. Hudec, E. Gramatová. An Efficient Functional Test Generation Method for Processors using Genetic Algorithms. Journal of Electrical Engineering JEEEC, ISSN 1335-3632, 2014, submitted to journal.