PHRT: A Model and Programmable Tool for Hardware Reengineering Automation

Oleg Nenashev
EDA Lab, Saint Petersburg State Polytechnical University
Russian Federation
nenashev@kspt.ftk.spbstu.ru

ABSTRACT

Hardware reengineering is a highly resource-consuming process of development cycle, so it is important to automate reengineering in order to reduce costs and provide reusable solutions. There are many specialized electronic design automation (EDA) tools for specific cases, but only few programmable tools supporting implementation of user-specific reengineering operations.

This paper presents PhD research, which aims development of such Programmable Hardware Reengineering Tool (PHRT), which can be useful for small hardware-design companies and research groups, who have specific recurrent tasks and cannot afford development of automation tools "from scratch".

We propose HDL-independent "hybrid" device representation model for automated analysis and transformation, which combines low-level structural descriptions(netlists) with features from high-level hardware description languages(HDLs). Such model supports parallel analysis and transformation of multiple description layers at once. In our research we present PHRT prototype, which is an extendable core, which provides basic functionality for import/export, analysis, editing and transformation of hybrid models. Its functionality can be extended by extensions and script programs.

At the current state, PHRT prototype is being successfully used by several Russian hardware-design companies. Test results have proven applicability of PHRT as a good framework for user-specific reengineering cases like testing instrumentation and reliability assurance (memory replacement, structural redundancy insertion, etc.).

Categories and Subject Descriptors

B.6.3 [**Hardware**]: Logic Design. Design Aids; D.2.7 [**Software engineering**]: Distribution, Maintenance, and Enhancement. Restructuring, reverse engineering, and reengineering; I.6.5 [**Computing Methodologies**]: Simulation and modeling. Model Development; J.6 [**Computer Applications**]: Computer-Aided Engineering

ESEC/FSE'13, August 18–26, 2013, Saint Petersburg, Russia

http://dx.doi.org/10.1145/2491411.2492406

General Terms

Theory, Design, Algorithms

Keywords

Hardware reengineering, electronic design automation, verification platform, extensible toolkit, hybrid device model, test insertion

1. INTRODUCTION

Existing EDA tools automate most often reengineering tasks: refactoring, optimization, test generation, etc. On the other hand, there are many uncovered user-specific tasks, in case of which automation can save many resources. It would be great to have a toolkits, which simplify automation of such tasks by providing programmable and customizable platform, which can be extended by user. We distinguish following features of such tool:

- Programmability - internal programming language
- Extensibility - extension capabilities (plug-ins, etc.)
- Integrability - tool can be integrated into hardware development environment
- HDL-independence - algorithm should be implemented only once for different input/output formats
- Functional completeness - tool should support all stages: analysis, transformation and verification

Most of existing tools support only one HDL language or netlist at once. In our research we focus on HDL-independent tools, because this tools allow to implement use-cases of hardware reengineering, which are not supported by existing tools. Examples:

- Parallel transformation of high-level description languages and theirs compilation(synthesis) results
- Migration between different hardware representations (ex, reverse engineering of netlists)

Therefore, development of universal PHRT is an actual problem, which requires specific device representation model and methodologies. Our work is devoted to development of model and tool for programmable hardware reengineering that should support HDL-independent device representation. This paper provides short overview of research project.

Section 2 contains brief overview of existing tools and device representation methodologies, which are oriented to hardware reengineering. Section 3 defines aims, objectives and methods of the research. We also discuss expected scientific and practical results. Section 4 briefly describes current status of research and its main results: "hybrid" device model, PHRT's architecture and prototyping results.

2. RELATED WORK

Existing hardware IDEs automate only several narrow-specific use-cases like source codes refactoring or primitive transformations. For example, SIGASI HDT and AMIQ DVT support renaming of components and extraction of component definitions [6]. AMIQ DVT supports primitive scripting language for programming of component renaming, but these tools still very far from universal PHRT [3].

On the other hand, we can use language-oriented tools, which have been originated from software engineering area. DMS Software Reengineering Toolkit provides programmable modification of different languages like C++, Java and VHDL [2]. This tool uses Abstract Syntax Tree (AST) as internal model and oriented for source code reengineering. However, it is difficult to apply such approach to hardware reengineering, because HDL source codes do not provide enough information about synthesized device.

Usage of universal modeling languages seems to be the most perspective approach (ex, UML-based RT-UML or MARTE tools [7]). Automation seems to be difficult because of model's complexity, but successful automation in one area should be applicable to others. Anyway, UML experience in Software Engineering area discovered many issues of this approach.

Regarding hardware representation, Wilsey et al. proposed a model of VHDL for analysis and transformation and then extended their model in further works [8]. Model includes primitive "static" structural description and "sequential" part, which describes both behavioral and non-synthesized semantics. In addition, authors proposed model reduction and modification approaches for analysis and transformation needs. This model has been partially used in SUAVE tool [1]. Unfortunately, model is oriented to a subset of single HDL, so it can't be used at universal tool.

Existing tools work well at their application areas, but issue with universal tool is still open.

3. RESEARCH DESCRIPTION

3.1 Aims and Objectives

Our main aim is to decrease reengineering efforts and improve designs quality via automation of user-specific hardware development flows. We're going to achieve this aim via development of extensible toolkit and its internal device representation model.

Development of the "hybrid" device representation model is a main objective of our research. Of course, we need to develop methodologies for model's import/export from existing HDLs, analysis, transformation, etc. Then, automation of reengineering use-cases provides opportunities for an additional research.

During research we develop prototype, which could be used for evaluation of proposed approaches and prototyping of reengineering use-cases. Regarding PHRT, we have following objectives:

- Develop and implement PHRT architecture, which should meet specified requirements
- Implement hybrid device model and basic functionality for its analysis and modification
- Implement extensions for import/export of HDL and netlist formats (EDIF,VHDL, etc.)
- Prototype PHRT-based automation for typical hardware reengineering use-cases
- Cooperate with third-parties in order to get external evaluation and feedback

3.2 Research Methods

Our topic allows to define research aims and objectives, so we use "constructive research" during development of hybrid model and its methodologies. Because there is a lack of third-party solutions to compare, we combine practical methods (formal analysis of requirements, prototyping and simulation) with case research and qualitative estimations. In addition, we use exploratory research approaches for evaluation of reengineering use-cases.

3.3 Expected Results

We are going to achieve following scientific results:

- Hybrid device representation model, which combines low-level descriptions(netlists) and features from high-level HDLs
- Methodologies for hybrid model import/export from different netlist and HDL descriptions
- Methodology for merging of netlist and HDL models of the same device
- PHRT-driven approach for test instrumentation of the complex Systems-on-Chip
- Approach for co-verification of software model and device prototype using PHRT

4. CURRENT STATE OF RESEARCH

At the current state, we have developed hybrid device representation model and its transformation and analysis methodologies. Then, we have implemented most of them in our PHRT prototype, which has been successfully tested against different reengineering use-cases. Prototype has successfully passed evaluation testing. Sections below briefly describe mentioned topics.

4.1 Hybrid Device Model

Internal device representation model is a key point for every reengineering application, because it should provide enough information to fundamentally different use-cases. For example, optimization algorithms requires low-level structural descriptions, which are not accessible in HDL. On the other hand, code refactoring requires only high-level device descriptions.

In the our research we propose "hybrid" model, which combines both low-level and high-level descriptions. Figure 1 presents main components of that model. Hybrid model extends low-level descriptions (netlists) by features from high-level HDLs and object-oriented languages.

Hybrid model supports following HDL features: groups, structural inheritance, static parameterization (generics), conditional generation, and PHRT-specific elements like references. At current state, we are working on the following model limitations:

- Limited support of behavioral descriptions due to model complexity requirements
- No support of user's metadata at basic model components (comments, etc.)
- Limited support of functions and macros (affects emerging languages like SystemC)

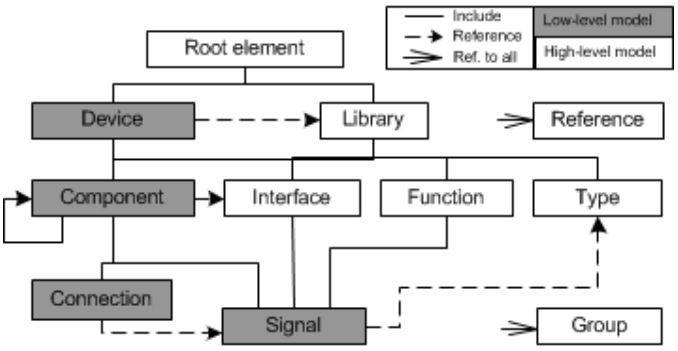

Figure 1: Elements of PHRT model

According to evaluation results, proposed PHRT model supports commonly used netlist and HDL formats (EDIF, VHDL, Verilog, SystemVerilog and SystemC). We suppose that it is possible to reuse "hybrid" model in order to perform reengineering of software. Thanks to SystemC support, model should support reduced C/C++ language subsets as well. Evaluation of common programming languages support is an additional topic for our research, because it will allow co-reengineering of system's software and hardware parts at once.

4.2 PHRT Prototype

PHRT is an extendable core, which provides basic functionality for import/export, analysis, editing and transformation of hybrid models described above. User-specific reengineering functions can be implemented in PHRT by plug-ins or external Tcl scripts. PHRT can be easily integrated with existing EDA tools from 3rd-party IC-design vendors via extensions.

Figure 4.2 shows two-layer architecture, which allows creation of user-specific tools based on PHRT core. First layer is internal core that provides API for custom extensions, this level is completely provided by our tool. Internal core and extensions together create user-specific core, which can be used by external environment.

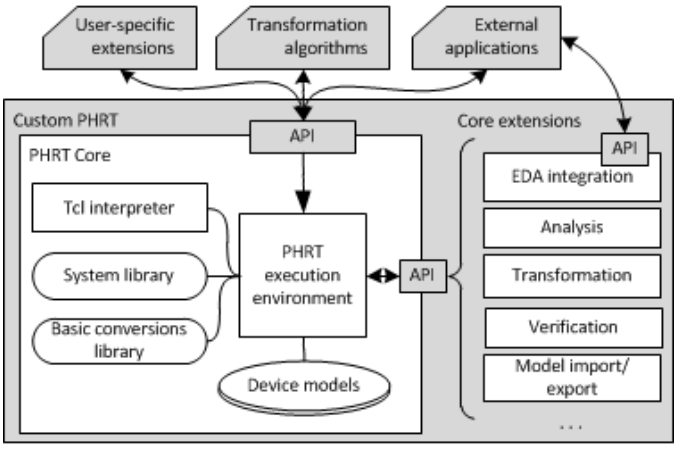

Figure 2: Extendable PHRT architecture

All previously implemented PHRT functions can be reused in more complex algorithms, so users can gradually extend functionality of their PHRT extensions in order to get automation of their reengineering processes in the future. During prototyping we have implemented extensions for the following purposes:

- Import/export of EDIF, VHDL and XML formats
- Reliability assurance: Structural redundancy insertion, memory blocks replacement [4]
- Analysis: Specific features like clock tree tracing or statistics about usage of library's elements
- In-circuit testing: Generation of testing and memory fault injection agents [5]
- Integration with QuartusII: Import/export of testing projects, conversion of behavioral descriptions
- User interfaces: Command console, model visualization, Eclipse IDE integration, etc.

4.3 Evaluation Results

During research we have implemented and successfully tested PHRT extensions for different use-cases. Section 4.2 provides list of available extensions. Currently, PHRT is being used at "Sitronics Microdesign"(Russian IC-design company) for real production needs. Our contributors have developed their own extensions in order to automate functional testing, design verification and memory fault injection (see section 4.4).

Evaluation results prove tool's applicability to different hardware development areas, but we have discovered several issues, which require modification of "hybrid" device model. So, we have a good proof of concept, but there is a lot work to do before public distribution of PHRT.

4.4 PHRT Use-Case Example

Below we provide short description of use-case from one of our contributors, who use PHRT in order to automate testing of their solutions at FPGA prototyping boards.

- Our contributors use their own tools to test designs on prototyping boards. These tools are not compatible with existing EDA tools.
- In order to run test, our contributors need to insert test infrastructure into their devices (ex, test agents and interfaces, which are specific to different tests).
- Each modification of initial design requires re-creation of all test infrastructure => such manual operation requires much efforts ($\tilde{5}$0% of overall testing time).
- PHRT completely automates generation of test infrastructure. It takes initial source codes, transforms device according to specified options and then outputs instrumented source code to synthesis tools.
- Therefore, user needs only to make several clicks in our PHRT prototype.

Described automation allows to spend much less time to routine operations. It means that test engineers will be able to spend more time to analysis and test coverage improvements.

5. CONCLUSIONS

We are sure that programmable hardware reengineering toolkits like PHRT could provide great assistance for developers, because they encourage development of reusable components and extensions for PHRT by reducing costs of component customization. Our prototype is a just first attempt to implement such tool. We are going to continue work on PHRT prototyping and its specific extensions in

order to expand tools' internal model and applicability.

Future research covers support of behavioral descriptions and software programming languages in "hybrid" model, built-in self-tests (BIST) insertion and other automation topics, which can be solved by PHRT in future. Current state of PHRT allows us to use in for prototyping during research projects in EDA and hardware development areas, so we are ready to cooperate with potential users of PHRT.

6. ACKNOWLEDGMENTS

I would like to thank my PhD advisor, professor Alexey Filippov from Saint-Petersburg State Polytechnical University, for guiding and supporting me during last years.

7. REFERENCES

[1] P. J. Ashenden, P. A. Wilsey, and D. E. Martin. SUAVE: Object-oriented and genericity extensions to vhdl for high-level modeling. In *Electronic chips & systems design languages*, pages 57–70. Springer, 2001.

[2] I. Baxter, C. Pidgeon, and M. Mehlich. DMS: Program transformations for practical scalable software evolution. In *Proceedings of the 26th International Conference on Software Engineering*, ICSE '04, pages 625–634, Washington, DC, USA, 2004. IEEE Computer Society.

[3] E. Linehan and S. Clarke. An aspect-oriented, model-driven approach to functional hardware verification. *Journal of Systems Architecture*, 2011.

[4] O. Nenashev. Developing a programmable toolkit for automated structural redundancy insertion. In *Best papers of XXXIX SPBSTU Week of Science*, volume 1, Saint Petersburg, 2011. SPBSTU.

[5] O. Nenashev. Automated test instrumentation of reliable digital devices for in-circuit testing. In *High-tech information systems and innovations in Russian National Research Universities*, volume 3, pages 64–68, Saint Petersburg, 2013. SPBSTU.

[6] B. Niton, K. Pozniak, and R. Romaniuk. Plug-in to eclipse environment for VHDL source code editor with advanced formatting of text. In *Proceedings of SPIE*, volume 8008, page 80080Q, 2011.

[7] J. Vidal, F. De Lamotte, G. Gogniat, P. Soulard, and J. Diguet. A co-design approach for embedded system modeling and code generation with uml and marte. In *Design, Automation & Test in Europe Conference & Exhibition, 2009. DATE'09.*, pages 226–231. IEEE, 2009.

[8] P. A. Wilsey, D. M. Benz, and S. L. Pandey. A model of vhdl for the analysis, transformation, and optimization of digital system designs. In *Design Automation Conference, 1995. Proceedings of the ASP-DAC'95/CHDL'95/VLSI'95., IFIP International Conference on Hardware Description Languages; IFIP International Conference on Very Large Scale Integration., Asian and South Pacific*, pages 611–616. IEEE, 1995.

APPENDIX

A. LIST OF PUBLICATIONS

A.1 Published

[1] O. Nenashev, "Automated test instrumentation of reliable digital devices for in-circuit testing" presented at the High-tech information systems and innovations in Russian National Research Universities, Saint Petersburg, 2013, vol. 3, pp. 64–68.

[2] O. Nenashev, "Concepts of programmable tool for automated analysis and transformation of digital device architectures described by VHDL" in Ist International Congress of PhD Students, Saint Petersburg, 2012.

[3] O. Nenashev, "Automated test instrumentation and BIST insertion with usage of automated reengineering toolkit" in Proceedings of XIIIth international conference "Fundamental and applied research in Russia" Saint Petersburg, 2012, vol. 2, pp. 46–49.

[4] O. Nenashev, "Developing a programmable toolkit for automated structural redundancy insertion" in Best papers of XXXIX SPBSTU Week of Science, Saint Petersburg, 2011, vol. 1, pp. 72–78.

[5] O. Nenashev, "Modification of Cypress WirelessUSB protocol in order to provide grid network topology" in Proceedings of XXXVIII SPBSTU Week of Science, Saint Petersburg, 2009.

[6] N. Baltrukov, O. Nenashev, A. Lavrov, and M. Kalugin, "Programmable Systems on Chip from Cypress Semiconductors" Saint Petersburg: SPBSTU, 2009.

[7] O. Nenashev, "Developing an adaptive control system for DC Motor" presented at the Proceedings of XXXVI SPBSTU Week of Science, Saint Petersburg, 2007.

A.2 Accepted Papers

[1] O. Nenashev, "PHRT: A programmable tool for automated hardware reengineering and verification" in Proceedings of the VES2013 workshop, Saint Petersburg, 2013.

A.3 Submitted Papers

[1] RAMS20014

[2] EWDTS2013