Kirill A. Bulygin
Year of birth: 1991
Faculty of Computer Technologies and Control Systems,
Department of Computer Science, group 6113
Speciality: 230100.68.13 “Embedded System Design”
Email: kirill.bulygin@gmail.com

UDC 004.4’244+004.436.2

TOOL FOR OBJECT-ORIENTED DESCRIPTION OF DIGITAL CIRCUITS

Kirill A. Bulygin
Supervisor: D.Sc., professor Alexey E. Platunov

The direction of research was methods to accelerate development of computer hardware. The research task was to develop a program that would simplify and speed up coding of digital circuits at the register-transfer level (RTL) [1]. In the study, it was determined that acceleration of computer hardware development can be achieved if developers are provided with the opportunity to use a simplified and extensible hardware description language. Descriptions written in this language can be processed (both translated and analyzed) by an intermediate EDA (Electronic Design Automation) tool.

It was also determined that the following features should be used in order to simplify extension of such a language:

- A hardware description should have an object-oriented intermediate representation, which requires comparatively a small amount of custom code (written in a scripting language) for processing (modification/analysis).
- The grammar of the language should be represented in one of the conventional Extended Backus-Naur Forms (EBNF).

Thus, the aim of this work was to create a user-extendable program that would allow working with hardware descriptions at the object-model level, accepting (as the input) RTL descriptions written in an intermediate (more convenient) language and generating RTL descriptions in a target language (supported by existing EDA tools). The basic requirements for the system are ability to use shorthand notations and simplicity of its extension.

According to this, Perl programming language has been chosen for development of the EDA tool, as this scripting language has advanced capabilities of text processing. An EBNF-compatible parsing library (Marpa::R2) has been chosen. To provide greater transparency and extensibility of the EDA tool, XML descriptions have been used to represent intermediate results of the processing.

The extensible EDA tool has been developed. It allows describing digital circuits using a simplified and user-extensible language. Output descriptions are represented in Verilog (which is one of the conventional hardware description languages).

As the input, the tool (Fig.) accepts the PDF document describing the grammar of Verilog HDL [2]. This description is extended and translated to the form supported by the parsing module (MarpaX::Import, which provides a convenient interface to Marpa::R2). Besides that, the tool reads hardware descriptions written both in Verilog and in the developed language named VT (Verilog Template). As the output, the tool generates Verilog modules.

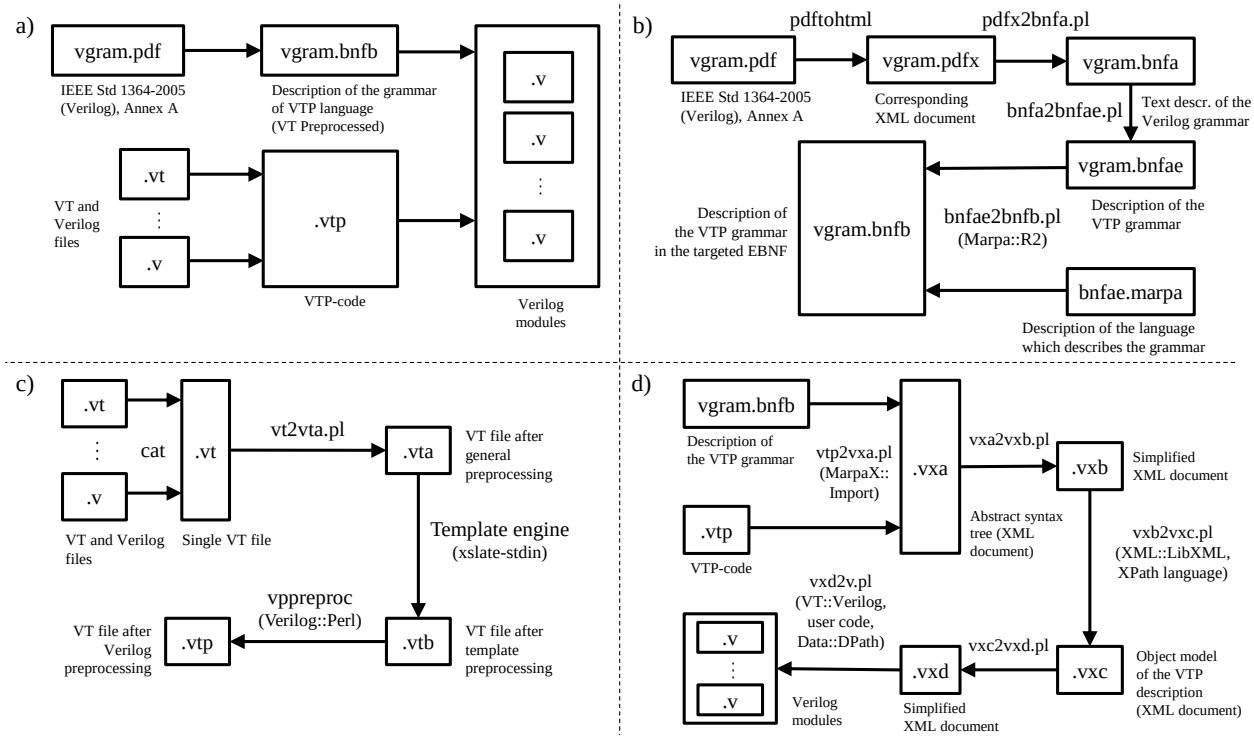

Figure: General processing diagram (a), generation of the description of the VTP grammar (b), translation of VT and Verilog modules to VTP code (c), translation of VTP code to Verilog modules (d)

Examples of the language features implemented (as a demonstration) through custom code are as follows:

- FSM registers (each of which corresponds to a pair of Verilog registers; the code to reset and maintain their state is generated automatically);
- contractions of parameter lists (e.g. lists of FSM states);
- shorthand notation for ranges;
- C-style statement brackets;
- compound operators for blocking assignment (e.g. “+=”);
- possibility to omit some keywords.

Processing on the object-model level has made it possible to add new features through quite a small amount of custom code written in Perl scripting language. For instance, the support of compound assignment operators was implemented with two blocks, of 6 lines of code (syntactic support) and of 11 lines of code (object-level processing).

The novelty of the result is that a user can extend the hardware description language and write a code to process descriptions at the object-model level, which simplifies the extension and allows both synthesis and analysis of hardware descriptions using a small amount of source code written in a scripting language.

The work is valuable primarily for developers of computer hardware who are familiar with Verilog. However, the described approach can also be applied to other specialized computer languages.

References

1. The Authoritative Dictionary of IEEE Standards Terms, 7th ed. — IEEE Press, 2000.
2. IEEE Standard for Verilog Hardware Description Language (IEEE Std 1364-2005). — IEEE Press, 2005.