Validating the PSL/Sugar Semantics Using Automated Reasoning

Michael J. C. Gordon
University of Cambridge Computer Laboratory, Cambridge, UK

Abstract. The Accellera organisation selected Sugar, IBM’s formal specification language, as the basis for a standard to ‘drive assertion-based verification’ in the electronics industry. Sugar combines regular expressions, Linear Temporal Logic (LTL) and Computation Tree Logic (CTL) into a property language intended for both static verification (e.g. model checking) and dynamic verification (e.g. simulation). In 2003 Accellera decided to rename the evolving standard to ‘Accellera Property Specification Language’ (or ‘PSL’ for short).

We motivate and describe a deep semantic embedding of PSL in the version of higher-order logic supported by the HOL 4 theorem-proving system. The main goal of this paper is to demonstrate that mechanised theorem proving can be a useful aid to the validation of the semantics of an industrial design language.

Keywords: Accellera; Formal verification; Higher-order logic; HOL; Model checking; Property language; PSL; Semantics; Sugar; Theorem proving

1. Background on the Accellera Organisation and the Sugar Property Language

The Accellera organisation’s website contains the following mission statement:

To improve designers’ productivity, the electronic design industry needs a methodology based on both worldwide standards and open interfaces. Accellera was formed in 2000 through the unification of Open Verilog International and VHDL International to focus on identifying new standards, development of standards and formats, and to foster the adoption of new methodologies.

Accellera’s mission is to drive worldwide development and use of standards required by systems, semiconductor and design tools companies, which enhance a language-based design automation process. Its Board of Directors guides all the operations and activities of the organisation and is comprised of representatives from ASIC manufacturers, systems companies and design tool vendors.

Faced with a growing number of syntactically and semantically incompatible formal property languages, Accellera set up a committee:

The Accellera Formal Property Language Technical Committee is chartered with the responsibility of defining a property specification language standard compatible with both the Verilog (IEEE-1364) and VHDL
Validating the PSL/Sugar Semantics Using Automated Reasoning

This formal language is targeted for both dynamic verification (e.g. simulation) as well as static verification (e.g. model checking). In addition, the Formal Property Language Technical Committee is chartered with:

- Driving standardisation among developers, users and academia,
- Promoting use of the standard, and
- Assuring interoperability for the property specification language among various verification tools within the hardware design flow.

This committee conducted a competition to select a property language design to be the basis of the Accellera standard. The finalists of the competition were based on four existing languages:

- Motorola’s CBV language;
- IBM’s Sugar (the language of its RuleBase FV toolset);
- Intel’s ForSpec;
- Verisity’s e language (the language of the Specman Elite test-bench).

After a combination of discussion and voting, the field was narrowed down to Sugar and CBV, and then in April 2002 a vote selected IBM’s submission, Sugar. The Accellera Formal Property Language Technical Committee then used Sugar as the starting point for defining a standard language they named PSL (short for Accellera Property Specification Language). This paper describes work that started with Sugar and then evolved to apply to PSL, and so we will sometimes refer to the language as PSL/Sugar.

The version of Sugar submitted by IBM to Accellera (Sugar 2.0) is primarily an LTL-based language that is a successor to the CTL-based Sugar 1 [BBE01]. A key idea of both languages is the use of extended regular expression constructs called Sugar Extended Regular Expressions or SEREs. Sugar 2.0 retains CTL constructs in its Optional Branching Extension (OBE), but this is de-emphasised in the defining document.

Besides moving from CTL to LTL, Sugar 2.0 supports clocking and finite paths. Clocking allows one to specify on which clocks signals are sampled (different sub-formulas may be evaluated with respect to different clocks). The finite path semantics allows properties to be interpreted on simulation runs by test-bench tools. The addition of clocking and finite path semantics makes the Sugar 2.0 semantics much more complicated than the Sugar 1 semantics. However, for a real ‘industry standard’ language Sugar is relatively simple.

The aim of this paper is both to document the representation of the semantics of PSL/Sugar in higher-order logic and also to describe how theorem proving has been used to validate the semantics. In Section 2, various motivations for this work are discussed. In Section 3, higher-order logic and semantic embedding are reviewed. In Section 4, the full semantics of PSL in higher-order logic is described. In Section 5, progress so far in analysing the semantics using the HOL 4 theorem-proving system is discussed. Finally, in Section 6 there is a short section of conclusions and future plans.

The material shown in framed boxes is copied from the LaTeX sources of the the Accellera Property Specification Language Reference Manual [Acc] (henceforth called ‘LRM’ for short). Note that, due to different style files used for this paper and the LRM, the text in the boxes will be typeset differently from how it is typeset in the LRM.

2. Why Embed PSL/Sugar in HOL?

There are various reasons for embedding PSL/Sugar in a machine-processable formal logic. The examples in the following sections are not meant to be exhaustive.

2.1. Debugging and Proving Meta-theorems

By formalising the semantics and passing it through a parser and type-checker one achieves a first level of ‘sanity checking’ of the definition. One also exposes possible ambiguities, fuzzy corner cases etc. The process is also very educational for the formaliser and a good learning exercise.

There are a number of meta-theorems one might expect to be true, and proving them with a theorem prover provides a further and deeper kind of sanity checking. In the case of PSL/Sugar, such meta-theorems include showing that expected simplifications to the semantics occur if there is no non-trivial clocking, that different