Chapter 19

COMBINED FORMAL REFINEMENT AND MODEL CHECKING FOR REAL-TIME SYSTEMS VERIFICATION

Alexander Krupp\textsuperscript{1}, Wolfgang Mueller\textsuperscript{1}, Ian Oliver\textsuperscript{2}

\textsuperscript{1}Paderborn University, Paderborn, Germany
\textsuperscript{2}Nokia Research Centre, Helsinki, Finland

Abstract We present a framework, which combines model checking and theorem prover based refinement for real-time systems design focusing on the refinement of non deterministic to timed deterministic finite state machines. Our verification flow starts from a cycle accurate finite state machine for the RAVEN model checker. We present a translation, which transforms the model into efficient B language code. After refining the RAVEN model and annotating it, the time accurate model is also translated to B so that the B theorem prover can verify the refined model almost automatically. The approach is introduced by the example of a mobile phone echo cancellation unit.

1. Introduction

With the increasing complexity of systems on a chip, formal verification is of increasing importance. In several fields of hardware design, equivalence checks and model checking are applied on a regular basis. For software design, the B method has been successfully applied in several industrial projects. Additionally, model checking receives increasing acceptance in embedded software design. However, theorem proving in general is less accepted since it still requires too many user interactions conducted by educated experts.

*The work described herein is funded by the IST Project PUSSEE 301

This article describes an efficient combination of model checking and theorem proven refinement based on the RAVEN model checker [13] and the Atelier-B B toolset [1]. We present a model checking oriented verification flow, which is based on automatic translation to B for refinement verification. We focus our interest on the refinement of a cycle accurate model into a time accurate model, which might be further refined to an implementation. In that context, we present a B code generation for a very efficient application of the Atelier-B theorem prover. In contrast to related approaches, our experimental results demonstrate that the proof of the generated code requires almost no interaction with the prover and additionally gives very low runtimes.

The remainder of this article is structured as follows. The next section discusses related works. Section 3 and 4 introduce RAVEN and B before Section 5 outlines our approach by the examples of the refinement of an echo cancellation unit. Thereafter, we present our experimental results before the final section closes with a conclusion.

2. Related Work

There is significant work integrating model checkers into theorem provers or vice versa. PVS (Prototype Verification System) is a theorem prover where the PVS specification language is based on high order predicate logic. Shankar et al. enhance PVS with tools for abstraction, invariant generation, program analysis (such as slicing), theorem proving, and model checking to separate concerns as well as to cover concurrent systems properties [14]. STeP (Stanford Temporal Prover) is implemented in Standard ML and C [9] and integrates a model checker into an automatic deductive theorem prover. The input for model checking is given as a set of temporal formulae and a transition system, which is generated from a description in a reactive system specification language (SPL) or a description of a VHDL subset. The SyMP framework integrates a model checker into a HOL based theorem prover for general investigations on effectiveness and efficiency [4]. The work focus on computer assisted manual proofs where main examples come from hardware design. Mocha [2] is a model checker enhanced by a theorem prover and a simulator to provide an interactive environment for concurrent system specification and verification. However, in the Mocha framework theorem proving is interactive and no efficient reasoning of more complex systems is reported.

In the context of the B theorem prover, Mikhailov and Butler combine theorem proving and constraint solving [10]. They focus on the B theorem prover and the Alloy Constraint Analyser for general property