Proving Hardware Designs

Peter T. Breuer    Luis Sánchez    Carlos Delgado Kloos
Departamento de Ingeniería de Sistemas Telemáticos,
Universidad Politécnica de Madrid, ETSI Telecomunicación,
Ciudad Universitaria, E-28040 Madrid, Spain
<{ptb,lsanchez,cdk}@dit.upm.es>

Keywords: Theorem proving applications; VHDL, hardware design validation.

VHDL is a standardized hardware description language with almost universal market penetration. Originally developed in the 1980s with the support of the US DoD, several formalizations of its hardware simulation semantics have appeared in the last few years, including operational semantics in higher order logic and translations to Petri Nets and finite state machines. But in recent work [1] we have set out the first simple, formal compositional denotational semantics for VHDL. It generates an axiomatic pre-/post-condition semantics. In classic style, when read as a PROLOG program, the logical rules run as a validation condition generator, an automated aid for correctness derivations.

The semantics is based on the idea that VHDL statements are pure side effects on a pair consisting of

- a bi-infinite sequence of historic and planned program states $W$, one for each moment of time;
- a pointer $T$ to the current time.

This design makes Hoare-style programming logic applicable. Importantly for safety-critical requirements on reliability, the rules code up directly in PROLOG:

$\{H_1 \vdash \text{Pre}, T_1\} \text{wait on } x \{H_2 \vdash \text{Post}, T_2\}$

$\vdash ...$

where $H_1$, $H_2$ gather lemmas to be proved. Giving $\text{Post}$ generates $\text{Pre}$.

References


Figure 1: The semantics of the wait on $x$ statement. No change in historic or planned states, but a forward shift in the current time pointer.