ABSTRACT

The design of digital devices now resembles traditional computer programming. Components are specified in a specialised form of programming language known as a hardware description language. Programs written in such languages are then executed to simulate the behaviour of the hardware they describe. These simulations cannot be exhaustive in most situations, so result in high, yet incomplete, confidence that the proper behaviour has been achieved.

The formal analysis of programming languages provides ways of mathematically proving properties of programs. These properties apply to behaviours resulting from all possible inputs rather just than a subset of them. The prerequisite for such an analysis is a formal understanding of the semantics of the language.

The Very High Speed Hardware Description Language (VHDL) is currently used to specify and simulate a wide range of digital devices. The language has no formal mathematical semantics as part of its definition, hence programs written in it have not been amenable to formal analysis.

The work presented here defines a structural operational semantics for a subset of VHDL. The semantics is then embedded in a mechanical proof assistant. This mechanisation allows one not only to reason about individual programs but also to express equivalences between programs. Examples which highlight the methodology used in this reasoning are provided as a series of case studies.