論文の内容の要旨

論文題目　Efficient Formal Equivalence Checking Methods for System-Level Design Descriptions
(システムレベル設計記述に対する効率的な形式的等価性検証手法に関する研究)

氏　名　松本 剛史

Due to the great advance of semiconductor technology, the integration of VLSI (Very Large Scale Integration) circuits has been increased for many years. This enables to integrate more and more transistors on a chip, which results in that a large whole system can be realized as a single chip so called System-on-a-Chip (SoC). When we design an SoC, it is a serious problem that the design period tends to be very long since the design size of SoCs is much larger than that of the conventional VLSI designs. To use billions of transistors that can be integrated in a chip, the design productivity of SoCs should be much improved.

One solution to improve the design productivity is introducing system-level design as a starting point of SoC design. In the most hardware designs, designers start designing from RTL (Register Transfer Level). In RTL, the function executed by combinational circuits for each clock cycle and the hardware resources (for example, adder and multiplier) to execute the function are decided. On the other hand, in system level, the (partial) execution order of the behaviors and the functional units consisting of many memory elements and combinational circuits (for example, filter and inverse discrete cosine transform) to execute the behaviors are decided. Therefore, we can say that system level design is more abstract than RTL design in terms of time and hardware resources. Since SoCs usually implement a system consisting of both hardware and software, it is preferable to use a single design language to describe designs. To satisfy this need, C language or C-based design language is used in system level design, which enables to design both hardware and software seamlessly.

Currently, system-level design is not completely automated by tools.

That is, before the final design descriptions that can be processed by behavioral synthesizers and compilers are generated, a number of refinements, changes, and optimizations of design descriptions are carried out by designers. Therefore, it is very important to check the equivalence of the design descriptions when they are modified. This is because, if the bugs inserted in system level are found in the later design steps, for example in RTL or in gate-level, a lot of time and cost will be spent to debug. Therefore, equivalence checking of system-level designs is studied.

In system-level design, there are few formal verification methods that are used widely in industry, and simulation plays a main role to verify designs. This is because large system-level designs cannot be solved by formal methods in practical periods. At the same time, however, simulation has also a serious problem in feeding good test patterns when a design is very large. Currently, state-of-the-art formal verification methods can solve one module of system-level design, which is corresponding to the size that can be synthesized by behavioral synthesizers. If our proposed methods can check the equivalence of two large design consisting of several modules, it can be said that the scalability of formal methods is much improved.

One powerful method to check the equivalence is applying symbolic simulation to both of the designs under verification with generating equivalence classes of variables and expressions. This approach does not need any test patterns, hence, can be classified as formal verification. However, it cannot be applied to large designs since the run time of symbolic simulation increases exponentially to the design sizes. To realize efficient equivalence checking of large system-level design descriptions, in this thesis, several verification methods are proposed.

The proposed verification methods utilize the difference between the design descriptions. In practical, system-level design is proceeded by gradually refining designs step by step. Therefore, the difference between designs of one refinement step is expected to be relatively small. The basic idea of the proposed method is that we can reduce the computation effort of equivalence checking utilizing the difference.

An efficient equivalence checking method is proposed that reduces the number of equivalence checking between variables and expressions utilizing difference. Using the difference and dependence of designs, in this method, equivalence checking of variables and expressions is not

carried out when two variables under checking are not affected by any difference. This method enables more efficient verification for designs with the same or similar control structures. However, the verification time by this method increases exponentially to the design size, although it can reduce a large number of equivalence checking between variables and expressions.

A more efficient equivalence checking method using difference is proposed. In the method, symbolic simulation is applied only to the related portions to the difference, while it is applied from the start to the end of each path in the previous method. For each difference between the designs under verification, equivalence checking based on symbolic simulation is performed first only for the difference. If all differences can be proved to be equivalent, the result of the verification is equivalent. When a difference cannot be proved to be equivalent, the verification is repeated extending the verification areas until the equivalence is proved. The extension of the verification areas is carried out along data and control dependence. This local checking approach results in that the equivalence can be proved with small computation effort even if the design itself is very large.

When verifying designs including parallel behaviors, it is impossible to apply equivalence checking to all possible schedulings, since the number of schedulings increases exponentially to the design size. To solve the problem, a sequentialization method is proposed. Given a design description with parallel behaviors, the method generates an equivalent design description without parallel behaviors. In the method, two statements that can be executed in parallel and dependent to each other are checked whether or not the execution orders of them is always the same. If the statements are always executed in the same order due to synchronization, they can be sequentialized into an equivalent two sequential statements. Otherwise, they cannot be sequentialized, since the executions of the statements may occur different results depending the execution orders. Using this sequentialization method, equivalence checking of two parallel behaviors can be reduced to equivalence checking of only a pair of two sequential behaviors, instead of checking many pairs of all possible scheduling.

When symbolic simulation is applied to loops, they are unrolled to avoid the execution paths with infinite length. This results in the increase of verification time especially when the number of unrolling is

large. As a solution of the problem, an equivalence checking method of loops without loop unrolling is proposed. It identifies the symbolic values of loop iterators that are required to compute an arbitrary index of the output arrays. After the required symbolic values of the iterators are extracted, symbolic simulation is applied only to the values for equivalence checking. As a result, the number of statements to be symbolically executed does not increase even if the number of iterations is actually large.

The several experiments conducted in this thesis confirm that the proposed methods enable to verify the equivalence of large system-level designs between practical design refinements.