Using Fixed-Point Semantics to Prove Retiming Lemmas

STEPHEN BROOKES
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213

Abstract. Algorithms designed for VLSI implementation are commonly described by directed graphs, in which the nodes represent functional units and the arcs indicate communication links. We give a denotational semantics for such a graph in terms of the least fixed point of a set of (mutually recursive) function definitions, describing the outputs produced at each node as a function of time. This semantics is consistent with the conventional clocked operational semantics of the system. A retiming is a systematic modification of the internode delays of a design, often used to convert an algorithm design into a systolic form. The utility of such retimings in optimizing the behavior of designs is well known. We use fixed-point semantics to provide simple proofs of the correctness of certain retiming transformations from the literature and to justify other design transformations such as pipelining.

Keywords: systematic arrays, fixed-point semantics, pipelining, retiming

1. Introduction

The rapidly increasing use of VLSI technology and the potential offered by VLSI technology for parallel computation are well known. It is therefore important to provide rigorous methods of specifying and proving correctness properties of algorithms designed for VLSI implementation. It seems to be common to use a high-level abstract model of algorithm designs in which a design is presented as a labeled directed graph whose nodes are regarded as standing for combinational units and whose edges represent dataflow connections. Such graphs (or related models) have been used (either explicitly or implicitly) by various people in specification and verification of the correctness of VLSI designs, often by using a semantic model in which the behavior of each node is modeled as a function (typically, from time to data values). For instance, functional models are used by Leiserson and Saxe [1], by Kung and Lin [2], in Chen’s work on space–time equations [3], in Chen and Mead’s hierarchical simulation [4], and in the work of Gordon [5].

In most of these earlier papers, correctness arguments were based (usually implicitly) on operational reasoning, analyzing the “state” of a system at each time step and arguing by induction on the number of elapsed time steps or clock cycles. This reliance on operational reasoning can result in rather complicated proofs [1]. In this article, we show that some elementary ideas from denotational semantics [6, 7] can be used to give simple correctness proofs. We focus on