Formal Derivation of a Loadable Asynchronous Counter *

Mark B. Josephs

Centre for Concurrent Systems and VLSI, South Bank University,
School of Computing, Information Systems and Mathematics,
103 Borough Road, London SE1 0AA, U.K. Mark.Josephs@sbu.ac.uk

Abstract. A loadable asynchronous counter is designed in a disciplined and formal manner. First, the behaviour of the counter is specified in terms of the way in which it handshakes with its environment. Next, the counter is decomposed into a linear array of cells, in which only adjacent cells can communicate, again by handshaking. Finally, the individual cells are decomposed into standard components that communicate by following a delay-insensitive signalling scheme.

In exploring the design space, "lazy cells" versus "eager cells" and "two-phase handshaking" versus "four-phase handshaking" are considered. Simplified designs are also provided for cells known to be always "even" and always "odd".

This example demonstrates how formal specification and verification can be carried out at appropriate levels of abstraction by using Handshake Algebra and Delay-Insensitive Algebra.

1 Introduction

This paper develops a solution to an asynchronous circuit design problem in a disciplined and formal way, using Handshake Algebra (HS-Algebra) [8, 5] and Delay-Insensitive Algebra (DI-Algebra) [6, 7]. It demonstrates how formal specification and verification can be carried out at the highest level of abstraction appropriate to each stage of design. In fact, this must be done if one is to keep the complexity of specification and proof under control.

Handshaking imposes a port structure on systems (asynchronous circuits in this case) and requires that the direction of communication on each port must alternate between input and output, something that is built into the semantics of HS-Algebra. We shall see how to define processes in HS-Algebra that specify the behaviour of components in terms of the way in which they handshake with their environment. Both two-phase handshakes and four-phase handshakes are shown to be representable within the formalism. The composition of components that handshake with each other will be calculated by symbolic manipulation.

Because handshaking is a restricted form of delay-insensitive signalling, processes in HS-Algebra can always be refined into DI-Algebra. This should be

* This work has been supported by the European Commission under Working Group ACiD-WG (Esprit Nr. 21949) as part of the Fourth Framework.
done when a component has been decomposed into parts as far as is practical at the handshaking level of abstraction; decomposition can then continue at a lower level of abstraction (top-down design). Conversely, processes in DI-Algebra can be abstracted into HS-Algebra by associating inputs and outputs with ports — this allows one to design bottom-up. Both refinement and abstraction will be illustrated in this paper. (As we shall also see, speed-independent or quasi-delay-insensitive signalling represents a still lower level of abstraction.)

In the remainder of this section, we overview the process algebras and provide the background to the design problem. The problem is concerned with systolic counters. Section 2 specifies a counter using HS-Algebra. Section 3 uses HS-Algebra to show how a \( k \)-bit counter can be decomposed into an array \( k \) of cells. Section 4 shows how, with DI-Algebra, we are able to design the counter cells out of standard asynchronous elements. The tradeoffs between “lazy” versus “eager” cells and between two-phase versus four-phase handshaking are indicated. Conclusions are drawn in Section 5.

**HS-Algebra and DI-Algebra**

HS-Algebra and DI-Algebra are formalisms that allow one to specify the way in which a system is required to communicate with its environment and to prove that a network of subsystems satisfies that specification. Specification involves the (recursive) definition of processes. Algebraic laws support reasoning about the equivalence and refinement of processes. Recursion-induction is available as a proof technique.

A complete set of laws for each algebra can be given (e.g. [1,8,12]), in the usual sense (cf. [16]) that every (non-recursive) process can be transformed into a normal form. Receptive process theory [3] provides a semantic model based on which the soundness of the laws can be demonstrated (e.g. [11]). Actually, much of the theory of Communicating Sequential Processes [2,16] carries over to HS-Algebra and DI-Algebra, including the notion of guardedness/constructiveness that is pertinent to proofs that involve recursion [13].

When defining a process \( P \) in HS-Algebra, we shall state its port alphabet \( pP \). Suppose \( c \in pP \). Then \( c? \) will denote an input signal, \( c! \) an output signal, \( c??! \) a passive (two-phase) handshake and \( c!?! \) an active handshake. It is remarkable that, while we can freely separate handshakes into their constituent signals, we can also express and manipulate handshakes as atomic operations within HS-Algebra.

On the other hand, a process \( P \) in DI-Algebra has both an input alphabet \( iP \) and an output alphabet \( oP \). These alphabets must be disjoint. As before \( c? \) will denote an input signal (but this time for \( c \in iP \) and \( c! \) an output signal (for \( c \in oP \))

More generally, signals and handshakes can involve communication of data. That is, \( c?x \) denotes a communication in which variable \( x \) is bound to the actual value input, and \( c!e \) denotes an output communication of the value of expression \( e \). Indeed, in the handshake \( c?x!e \) the variable \( x \) may occur in \( e \).