Available online at www.sciencedirect.com

SCIENCE DIRECT®

Information and Computation 204 (2006) 1295–1324

Information and Computation

www.elsevier.com/locate/ic

Synchronous structures ☆

David Nowak

Research Center for Information Security, National Institute of Advanced Industrial Science and Technology, Akihabara Daibiru, Room 1102, 1-18-13 Sotokanda, Chiyoda-ku, Tokyo 101-0021, Japan

Received 3 November 1999; revised 21 May 2001
Available online 12 June 2006

Abstract

Synchronous languages have been designed to ease the development of reactive systems, by providing a methodological framework for assisting system designers from the early stages of requirement specifications to the final stages of code generation or circuit production. Synchronous languages enable a very high-level specification and an extremely modular design of complex reactive systems by structural decomposition of them into elementary processes. We define an order-theoretical model that gives a unified mathematical formalisation of all the above aspects of the synchronous methodology and characterises the essentials of the synchronous paradigm.

Keywords: Preorder semantics; Reactive system; Synchronous programming

☆ A preliminary version of this article was presented at the 10th International Conference on Concurrency Theory (CONCUR'99), Eindhoven, The Netherlands, August 1999 [14]. This work was partially supported by Action Coopérative INRIA PRESYSA. This work was done while the author was affiliated with Institut de Recherche en Informatique et Systèmes Aléatoires (IRISA), Rennes, France, and Oxford University Computing Laboratory, United Kingdom.

E-mail address: nowak@lsv.ens-cachan.fr.

0890-5401/$ - see front matter © 2006 Elsevier Inc. All rights reserved
doi:10.1016/j.ic.2001.05.001

1. Introduction

1.1. The synchronous paradigm

Synchronous languages, such as Esterel [5], Lustre [10], and Signal [4] have been designed to ease the development of reactive systems. The synchronous hypothesis provides a deterministic notion of concurrency where operations and communications are instantaneous. In a synchronous language, concurrency is meant as a logical way to decompose the description of a system into a set of elementary communicating processes. Interaction between concurrent components is conceptually performed by broadcasting events. Synchronous languages enable a very high-level specification and an extremely modular design of complex reactive systems by structurally decomposing them into elementary processes. The use of synchronous languages provides a methodological framework for assisting the users from the early stages of requirement specifications to the final stages of code generation or circuit production while obeying compliance to expressed and implied safety requirements. In that context, the synchronous language Signal is particularly interesting, in that it allows the specification of (early) relational properties of systems which can then be progressively refined to obtain an executable specification. All the stages of this design process can easily be modelled and understood in isolation. The purpose of this article is to define the mathematical model of *synchronous structures* which gives a unified formalisation of all the aspects of a synchronous methodology and which contains each of them in isolation.

1.2. Related work

There are several ways to characterise the essentials of the synchronous paradigm. In [13], we introduce a co-inductive semantics of Signal, and a library of theorems is developed in the proof assistant Coq [18]. But it is not expressive enough to deal with dependencies. The semantics of a synchronous language can be described in a better way with Symbolic Transition Systems (STSs) [15]. This is a formalism on which fundamental questions can be investigated. But it treats the absence of a signal as a special value. This is not consistent with reality: The presence or the absence of a signal, relatively to another signal, has to be inferred by the program (endochrony [3]). In [3], STSs are extended with preorders and partial orders to model causality relations, schedulings and communications. This preorder-theoretic model is put into practice in the design of BDL [17], a synchronous specification language that uses families of preorders to specify systems. In [7], the problem of characterising synchrony without using a special symbol for absence is addressed in terms of multiple input–output sequential machines. In [8], the language Signal is modelled in interaction categories [1] where morphisms are processes and objects are types of processes.

1.3. Motivations

In 1545, the great Italian mathematician Gerolamo Cardano wrote “Ars Magna” [6], an important and influential treatise on Algebra in which the first complete expression for the solution of a general cubic equation was put forward. Cardano noticed that, in the case of some equation with three real solutions, he was forced to take at a certain stage the square root of a negative number. The imaginary numbers were born. Analogically, we generalise the classical notion of signal