I. Introduction

To guarantee the correctness of hardware/software systems, formal verification provides an alternative approach to methods such as test, or simulation. In that framework, one can either formally verify that an implementation of the expected device/program fulfills its specification, or transform the initial specification by stepwise correctness-preserving refinements. In both cases, an informal specification does not suffice. This is the reason why formal languages or methods, such as Z, VDM, B, CSP, CCS are becoming so vital in today's world where the importance of specifying and validating distributed architectures/programs, concurrent and reactive systems, communication protocols, is increasing every day.

We are developing a specification and proof environment, called PREVAIL [1], which is to support several input languages (currently, only VHDL is supported) and which proposes a set of proof tools to verify appropriate descriptions/specifications. Nthm [2] is one of these tools, and we are working at defining an induction-based method to validate concurrent systems using this prover. To give such systems a formal specification, our first task was to choose between VHDL [4] and one of the formal languages that can be of interest to hardware/software developers.

Using a simple but significant reactive system as running example, we give a comparative evaluation of VHDL and two formal languages : the ISO Formal Description Technique LOTOS [5], and the computational model UNITY [3]. Thus we draw conclusions about the accuracy of each one of them with respect to different aspects (sequential behaviours, communications, non determinism, fairness, etc...). More details can be found in [6].

II. Comparative evaluation

The reactive system that has been chosen as example is an elevator in a three-floor building. The system is composed of an operative part (the elevator) and of a control part (the controller). There are six buttons, three internal buttons and one button at each floor. Upon request, and according to the current state of the elevator, the controller sends an appropriate signal, and also indicates the direction priority (signal UP). The elevator informs the controller about its current state.

To our mind, this example is interesting because it illustrates several important concepts :
- concurrent processes and their communications : the processes Elevator and Controller exchange data through communication ports,
- communication with the environment : the process Controller inputs data from the outside world through input ports (E1, E2, E3, Req1, Req2, and Req3),
- determinism versus non-determinism : the system is deterministic. Non-determinism is a feature of LOTOS and UNITY, this example shows how deterministic systems can be specified using these formalisms. Conversely, this description is straightforward in VHDL. The description of a non-deterministic example would have been more tedious.

References


1066-1409/96 $5.00 © 1996 IEEE