PLC-Automata: A New Class of Implementable Real-Time Automata*

Henning Dierks**

University of Oldenburg, Germany

Abstract. We introduce a new class of automata which are tailored for dealing with real-time properties modelling the behaviour of Programmable Logic Controllers (PLC) that are often used in practice to solve controlling problems. A semantics in an appropriate temporal logic (Duration Calculus) is given and an implementation schema is presented in a programming language for PLCs that fits the semantics. Finally, a case study shows the suitability of this approach.

1 Introduction

In this paper we propose a language to specify real-time systems that fits both the needs of computer scientists and programmers of such systems. Formal specification and verification of real-time systems that are used in practice depend on the communication between the scientist who models the behaviour of the system by formal methods and the programmer who is working in practice with it.

This language is motivated by the experiences we made in the UniForM-project [7] with an industrial partner. The aim of the project is the development of real-time systems in a workbench using combinations of formal methods. We present a formal semantics that allows formal reasoning and proving correctness using the Duration Calculus [4] which is a suitable temporal logic. We also give an implementation of such systems in a particular hardware called Programmable Logic Controllers (PLC).

These PLCs are very often used in practice to implement real-time systems. The reason is that they provide both convenient methods to deal with time and an automatic polling mechanism. Nevertheless, every computer system can be used to implement the proposed language if a comparable handling of time and an explicit polling is added.

Furthermore, the language can be viewed as a definition of a very small but implementable subset of Timed Automata [1].

* This research was partially supported by the German Ministry for Education and Research (BMBF) as part of the project UniForM under grant No. FKZ 01 IS 521 B3.
** e-mail: dierks@informatik.uni-oldenburg.de
1 See the appendix.
2 The Behaviour of Programmable Logic Controllers

Programmable Logic Controllers (PLC) are often used in industry for solving tasks calling for real-time problems like railway crossings, traffic control, or production cells. Due to this special application background PLCs have features for making the design of time- and safety-critical systems easier:

- PLCs have input and output channels where sensors and actuators resp. can be plugged in.
- They behave in a cyclic manner where every cycle consists of the following phases:
  - Polling all inputs and storing the read values.
  - Compute the new values for the outputs.
  - Update all outputs.

The repeated execution of this cycle is managed by the operating system. The only part the programmer has to adapt is the computing phase. Thus, PLCs are implemented polling machines realising the typical method of solving time-critical problems in reality.

- Depending on the program and on the number of inputs and outputs there is an upper time bound for a cycle that can be used to calculate the reaction time.
- Convenient standardised libraries are given to simplify the handling of time.

Although these characteristics are quite useful PLC-programmers have to face the following problem: If an input signal does not hold for at least the maximum amount of time needed for a cycle, one cannot be sure that the PLC will ever read this signal. This problem can be solved either by

- changing the sensors used in the setting or by
- using PLCs that are fast enough.

The decision in which way the problem should be solved depends on availability and costs of both faster PLCs and sensors that assure longer lasting signals.

Another important feature of PLCs is that they can be coupled: the output of one PLC can be the input of another PLC. In fact, their operating systems do not differentiate between a sensor's input and a PLC's input and between an output to actuators or to PLCs respectively. Thus, the programmer is again obliged to consider how long an output signal from one PLC will be held and how long it must be held to make sure that it has been noticed by the other PLC.

Note that this is one advantage of using PLCs. It obliges the programmer to check both his sensors and cycle time, which makes the assumptions concerning the hardware explicit.