uRT51: AN EMBEDDED REAL-TIME PROCESSOR IMPLEMENTED ON FPGA DEVICES

R. CAYSSIALS†, M. DUVAL‡, E. FERRO† and O. ALIMENTI†

‡ Dpto. Ing. Elect., Univ. Nac. del Sur –CIC, 8000 Bahía Blanca, Argentina duval@criba.edu.ar

Abstract — In this paper we describe and evaluate the main features of the uRT51 processor. The uRT51 processor was designed for embedded real-time control applications. It is a processor architecture that incorporates the specific functions of a real-time system in hardware. It was described using synthesizable VHDL and it was implemented on FPGA devices. We describe how the uRT51 processor supports time, events, task and priorities. The performance of the uRT51 processor is evaluated using a control application as a case study. The experiments show that the uRT51 processor scheduling features outperform the ones obtained using a traditional RTOS-based real-time system.

Keywords — Scheduling, Real-Time, Embedded Processor

I. INTRODUCTION

Real-time systems are ubiquitous and diverse, and include mobile phones, entertainment devices, automobiles, toys, smartcards, medical devices, network switching equipment, sensors and industrial robots. They are modeled as a set of tasks that has to be executed by a processor before a certain deadline.

Embedded real-time systems are usually built using either Real-Time Operating Systems (RTOS) or designing the whole system as a monolithic program based on specific hardware resources. Real-time operating systems allow designer to achieve a higher abstraction level, higher portability and better verification and maintenance features of the system. Several RTOSs were proposed and designed in order to give real-time support to applications (e.g. POSIX.1003 (IEEE1003.1d, 1999), Spring kernel (Stankovic and Krithi, 1991), QNX (Hildebrand, 1992)). The RTOS includes a special system task, named the scheduler, which shares the processor among the tasks that require execution. The scheduler implements a priority discipline to determine the next task to be executed. The scheduler should be executed periodically and its execution produces overhead on the system.

Some applications may require stricter temporal features or may not tolerate the runtime overhead produced by the operating system. In these cases, a direct use of the hardware resources is mandatory. Timers, counters and interrupts have to be directly programmed avoiding an easy maintenance of the system. In addition, programming directly the hardware resources of the system may generate failures hard to debug.

On the other hand, several approaches have been proposed to implement processors with real-time features. In Halang and Colnaric (1993), Halang et al. (1995) and Glavinic et al. (1999), a co-processor architecture is proposed to support real-time applications. In Adomat et al. (1996), a real-time kernel is implemented in hardware. In Kohout et al. (2003), a Real-Time Task Manager (RTM) is proposed as a parallel unit to carry out the real-time behaviour of the processor. These architectures are very restrictive on both the priority discipline implemented and the number of tasks supported.

In Matthew et al. (1999), it is described the AAMP processor developed by Rockwell-Collins to be used in critical avionics applications. The AAMP was verified to have a predictable behaviour, but it does not have special real-time features but a multitasking support. None of these approaches are available for commercial, industrial or educational developments.

The uRT51 is a processor for embedded real-time applications. Its architecture was designed from a real-time point of view and it may support a great deal of priority disciplines over a set of up to 65,000 tasks. The uRT51 architecture is scalable and consequently the number of tasks and events supported depends on the amount of physical memory of the system. It is described in VHDL and it can be used to build system-on-chip (SoC) architectures over FPGA devices. With the uRT51 Real-Time Suite, temporal parameters can be defined independently of the functionality of each task. Hence, task’s code does not include real-time programming which permits a higher abstraction level of implementation and a better flexibility in the design (uRT51, 2005).

In this paper, we describe the main features of the uRT51 processor. We evaluate the performance of the uRT51 processor in a control application. We propose a speed control of a DC motor to measure the performance of the uRT51 processor. The experiments show that the uRT51 is much more efficient than an RTOS-based embedded real-time system.

This work was supported by the Technological Modernization Program under Grant BID1201/OC-AR.