Real-Time System Development in Ada using LEGO® Mindstorms® NXT

Peter J. Bradley
Real-Time Systems and Telematic Services Architecture Group
Universidad Politécnica de Madrid
Madrid 28040, SPAIN
pbradley@datsi.fi.upm.es

Juan A. de la Puente
Real-Time Systems and Telematic Services Architecture Group
Universidad Politécnica de Madrid
Madrid 28040, SPAIN
jpuente@dit.upm.es

Juan Zamorano
Real-Time Systems and Telematic Services Architecture Group
Universidad Politécnica de Madrid
Madrid 28040, SPAIN
jzamora@datsi.fi.upm.es

ABSTRACT
In this paper, we describe a set of tools to fully develop a real-time application under Linux using as target the LEGO® Mindstorms® NXT robotics kit. These tools provide Real-Time & Embedded systems teachers with an alternative to conventional software models designed in classrooms and labs.

Keywords
Real-Time & Embedded systems education, Ada, Ravenscar, LEGO® Mindstorms®.

1. INTRODUCTION
The experience gained by students in real-time system development in classrooms and labs has been frequently reduced to software models which lack a “real” development experience. Economic and complexity drawbacks of physical devices have been outmatched in the past years with simple and flexible programmable robotics kits like LEGO® Mindstorms®.

Our purpose is to provide a set of tools, so that master students of Embedded & Real-Time Systems at the UPM are able to analyse, develop and implement a real-time system in Ada for LEGO® Mindstorms® robots complying with the Ravenscar profile [3].

This set of tools includes (see figure 1):
• A cross compilation system including compiler, assembler, linker, etc...
• An API to interact with LEGO® Mindstorms® peripherals
• A custom firmware for LEGO® Mindstorms®
• Communication tools between the target and the host to upload firmware and programs
• A remote debugging system or JTAG hardware debugging for LEGO® Mindstorms®
• A WCET tool to procure a timing analysis of the system
• A real-time modelling tool to evaluate the feasibility of the system

All of these tools will be running under Linux as most of them are available for Linux. Besides, it's the default OS used in our labs for software development.

Figure 1. Development process.

2. TOOLS DESCRIPTION
The LEGO® Mindstorms® NXT [4] kit provides all the basic hardware features for classroom real-time system development. This is: an ARM7™ main processor, USB & bluetooth communications, I/O ports (motors, sensors …), and a considerable amount of LEGO® bricks to build complex models. Additionally, it offers support for a large variety of programming languages, which include Ada. There is a respectable and active research internet community.

Ada’s concurrency and real-time integrated features as well as its hardware and interrupt handling support makes it the ideal choice for real-time system development.

The first step, once the physical device (LEGO® Mindstorms® NXT) and the programming language (Ada) have been established, is to build a cross compilation system hosted in Linux for an “arm-elf” target. AdaCore’s GNAT GPL for LEGO® Mindstorms® 2009 [1] is hosted in Windows so a porting to Linux is needed. The packages to build include GNU Binutils (assembler, linker …), GMP & MPFR libraries (floating-point features), Newlib library (specific for embedded systems), gcc
compiler and GNAT front end. The GNU build system (Autotools) is used throughout the building process.

The resulting compilation system uses the zero footprint runtime and has no tasking features. To provide tasking, the Ada API library supplied, relies on nxtOSEK [7] (OSEK [8] compliant system for LEGO® Mindstorms®). nxtOSEK depends on a modified LEGO® firmware (nxtBIOS or John Hansen's enhanced firmware [2]).

Once the cross compilation system is working on Linux, students are able to build their own real-time software for the LEGO® Mindstorms® using the Ada API. The way to upload software and firmware to the LEGO® Mindstorms® NXT from a Linux system is to use John Hansen's NeXTTool [2] or LibNXT [5].

In order to study the feasibility and scheduling aspects of the generated real-time application, a MAST model will be prepared by the students. MAST [6] is an open source set of tools, developed at Universidad de Cantabria, that enables modelling real-time applications and performing timing analysis of those applications. The MAST model can be used to represent real-time behaviour and requirements allowing an automatic schedulability analysis.

Figure 1 shows an overview of the development process.

3. SAMPLE APPLICATION

As proof of concept, a wired controlled vehicle will be designed, see figure 2. This vehicle has a front castor wheel used to turn and two back wheels, each with an independent motor. To control the vehicle we will use a wired joystick made up of a touch sensor to start/stop propulsion and a motor’s encoder to control turns. Depending on the angle of the joystick encoder, different speed values will be send to the vehicle motors, thus turns are made possible, see figure 2.

First, students must analyse the problem and come up with a software architecture. During this step, tasks, shared resources, events and relations between them, must be identified, see figure 3.

After, students must code the final design to Ada. Since a WCET tool hasn’t yet been integrated, the physical model must be constructed in order to estimate the timing analysis of the system. This is done using the bluetooth API.

Finally, a MAST model is developed to study the feasibility and scheduling aspects of the vehicle project.

4. CONCLUSIONS & FUTURE WORK

We believe that a good foundation has been established for real-time & embedded system teachers to enlighten students in the development of “real” embedded platforms using Ada. The LEGO® Mindstorms® NXT kit offers all kinds of sensors and mechanisms to work with, even custom-made sensors can be...
developed. Also, there is a large and active internet community.

However, programming applications in Ada with the ZFP runtime, relying on nxtOSEK to achieve tasking and concurrency, is still far from ideal since no Ada tasking or protected objects features are accessible. Once AdaCore releases GNAT GPL for LEGO® Mindstorms® 2010 with Ravenscar runtime (due Sept. 2010) there will be no need to use nxtOSEK and all Ada features that comply with the Ravenscar profile will be operative. As with the 2009 version, a Linux port will be made accessible from our web page1.

Future work could adapt a WCET tool like RapiTime [10] or Bound-T [9] to our development process. Remote debugging could be achieved using a TCP based communication between LEGO® Mindstorms® and gdb via USB connection. Alternatively, hardware debugging using JTAG is also possible.

5. REFERENCES

1 http://polaris.dit.upm.es/~str/index.html