AC 2011-2476: THE VU-LEGO REAL TIME TARGET: TAKING STUDENT DESIGNS TO IMPLEMENTATION

James Peyton Jones, Villanova University

James Peyton Jones is Director of the Center for Nonlinear Dynamics & Control and Professor of Electrical & Computer Engineering at Villanova University

Connor W McArthur, Villanova University

Connor McArthur is an undergraduate at Villanova Unversity studying Computer Engineering and Computer Science.

Tyler A Young, villanova University

Tyler Young is a senior Computer Engineer and research assistant at Villanova University.

The VU-LEGO Real Time Target: Taking Student Designs to Implementation

J.C. Peyton Jones, C. McArthur, T. Young
Center for Nonlinear Dynamics & Control, Villanova University

Abstract

The use of embedded / mechatronic systems in teaching is being revolutionized by a) the advent of increasingly powerful yet low-cost computational devices and sensors, and b) by modern Automatic Code Generation tools which allow these devices to be programmed directly from high-level designs - without the difficulties traditionally associated with low level embedded system programming. This paper describes progress on a National Science Foundation, MathWorks and Nokia sponsored project aimed at exploiting these developments for practical use and benefit in the classroom. Specifically, the paper describes a new toolchain which enables students to access the hardware capabilities of the 32-bit LEGO NXT brick from within the Matlab / Simulink environment, and to automatically generate and cross-compile the necessary code for real time autonomous implementation. LEGO hardware I/O is represented in the Simulink design mode as blocks for accessing motors, encoders, push-buttons, ultrasound sensors, light sensors and more. Blocks have also been developed to allow the LEGO target to communicate in real time with a host computer over USB or BlueTooth communications. The toolbox enables students to undertake more complex and challenging problems while focusing on high-level pedagogical goals rather than low level issues. A series of tutorial examples are discussed.

1. Introduction

The use of embedded / mechatronic systems in teaching is being revolutionized by the advent of increasingly powerful computational devices and sensors, and by high-level Automatic Code Generation tools with which to program them. Low-cost yet highly capable computational hardware and sensors have evolved as a natural outgrowth of Moore's law, and the use of embedded computing devices has become nearly ubiquitous. High-profile examples include the Mars Rover and the fully autonomous vehicles of the DARPA Urban Challenge, but there are also many humbler examples encountered daily in the home. The family car, for example, has traditionally been regarded as a predominantly mechanical system, but today a typical car incorporates more than 50 embedded microprocessors, communicating with each other over in-vehicle networks and optimizing the overall system behavior with respect to safety, performance, fuel economy and emissions.

Embedded systems also have a well established track record in education, particularly in the robotics area. Many researchers have found that student's motivation to learn increases significantly with hands-on robotics-based projects, [1-3]. Others have successfully used

robotics as a unifying theme in introductory courses, [4-7], and still others have used robotics as way to attract women into Computer Science, [8]. The increase in embedded computational power, however, is also strongly correlated to programming complexity. This has motivated a move, both in academia and industry, away from low-level programming in assembly language, towards higher level programming in C, or still higher design environments such as Matlab and Simulink. In particular, modern Automatic Code Generation or 'Rapid Prototyping' tools allow embedded target devices to be programmed directly from high-level system designs without the traditional difficulties associated with low level issues, fixed point computation, or C-language programming. These tools also have the potential to transform the use of robotics or other embedded applications in education, enabling students to undertake more complex and challenging problems while focusing on the high-level pedagogical goals rather than low level issues.

In this paper, a new rapid prototyping toolbox, the Villanova University LEGO Real Time target (VU-LRT) is presented. The toolbox enables high-level designs coded in the Matlab/Simulink environment to be automatically cross-compiled for execution on the low-cost but remarkably capable LEGO MindStorms NXT brick. The paper is organized as follows. Section 2 discusses target hardware selection, as well as software alternatives, rapid prototyping tools and the choice of the Simulink design environment for this project. The specifics of the VU-LRT toolbox are presented in section 3, and a brief example is given in section 4. Finally brief conclusions and plans for future work are discussed in section 5.

2. The LEGO MindStorms NXT: Hardware and Software Alternatives

The evolution of embedded computing devices is reflected in the wide variety of robot hardware platforms in use within the STEM community. Most of these devices, such as the Parralax BOE Bot [9], HandyBoard [10], ActivMedia [11], Arduino [12] and first generation LEGO RCX brick [13] for example, are still based on 8-bit processors. Typically these machines run at clock speeds of 20 MHz or less, have 32 KB or less of RAM and are hard to program effectively because of finite word length and memory issues. Recently, however, 32-bit machines have become available at very similar cost to their 8-bit predecessors. These include the LEGO Mindstorms NXT [14], the Surveyor SRV1 [15], and the Korebot, [16]. The LEGO Mindstorms NXT, for example, is a 32-bit machine with twice as much memory as the older RCX version (Fig. 1). It has Bluetooth communications which allow a master unit to communicate with up to three secondary NXT units, and features 4 input ports and 3 output ports. Users can also access the in-built push-buttons and on-board loudspeaker. The output ports can be used to drive *dc* motors and incorporate encoder position feedback, and the input ports can be connected to an increasingly broad variety of sensors available from LEGO MindStorms or from third party suppliers such as HiTechnic and Vernier Software & Technology. Finally, the LEGO MindStorms NXT integrates seamlessly into the LEGO Technic world, enabling students to implement both the software and mechanical / hardware aspects of their designs. The LEGO MindStorms NXT was therefore chosen as the hardware platform for this project.

As outlined in the introduction, advances in embedded hardware have also motivated advances in tools for software development, and a variety of high level design tools are now available. To varying degrees, these tools allow users first to simulate their designs, then implement them on

Figure 1. The LEGO MindStorms NXT brick and associated peripherals

target hardware, and finally to tune system parameters while the code is actually running on the target. This development cycle is both practical and educational and is widely used in industry. Specifically, these tools include MicroSoft Robotics Studio (MSRS), LabView from National Instruments, and Matlab / Simulink from the Mathworks. The Matlab / Simulink environment which is arguably the most pervasive in the STEM community, is already tightly integrated into the research activities and educational curriculum at Villanova University and other institutions. Simulink was therefore chosen as the design environment for the project.

The use of Matlab and Simulink for educational robotics applications is not new. Dr. Behrens, from the Institute of Imaging and Computer Vision, in Aachen, Germany developed the RWTH toolbox for wirelessly sending commands and receiving data to/from a LEGO NXT platform [17]. This 'remote control' approach has the control algorithm running on the host PC with the robot acting primarily as a dumb sensor / actuator. Though simple, the approach is limited to low bandwidth control applications due to the time varying delays which inevitably occur in the host-target communication channel. A truly embedded / real-time solution, very similar to that advocated in this paper, has also been developed by T. Chikamasa in the form of the Embedded Robot (ECRobot) coder [18]. The ECRobot toolbox represents a significant advance from hand-coded algorithm implementation and has been used by the authors and others in earlier projects [19,20]. However its function-call based architecture does not conform to the normal Simulink Real-Time Workshop design process, and it imposes significant constraints on user designs. The VU-LRT toolbox aims to provide a more user-friendly blockset and to integrate more seamlessly with the standard RTW design process. However, it still builds on the same real-time operating system used by ECRobot and has benefitted considerably from this earlier work.

3. The Villanova University LEGO Real Time target (VU-LRT)

The Villanova University LEGO Real Time target (VU-LRT) provides a blockset and toolchain to enable target implementation of high-level Simulink designs on the NXT brick. The blockset defines a high-level interface to NXT hardware for users, as well as the target-specific low-level code and cross-compilation details necessary to implement this in the final executable. The non-target-specific parts of the code generation and the overall 'build' process are handled by the

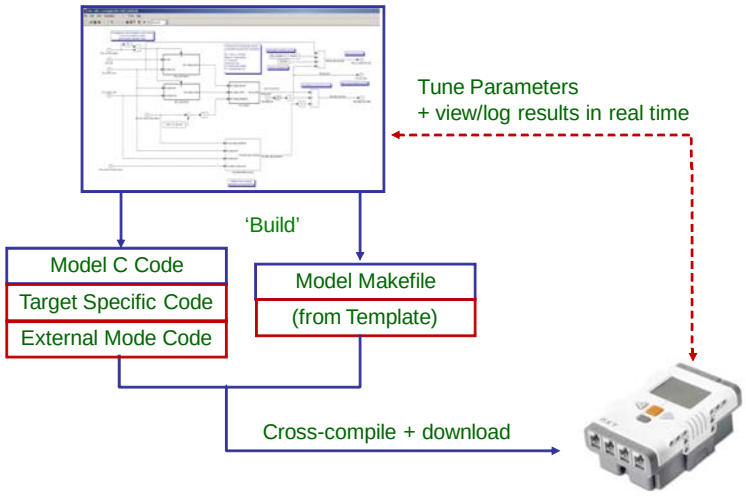

Figure 2. The Rapid Prototyping Process

MathWorks Real Time Workshop (RTW) toolbox. The process is illustrated in Fig. 2. The start point is a user design in the form of a Simulink model. When the user initiates a 'build' command, the Real Time Workshop automatically generates the corresponding C code, as well as a 'makefile' which defines how to cross-compile this code into a real-time executable. The VU-LRT toolbox provides the NXT-specific template that is used by the Real Time Workshop to generate the makefile, as well as the target specific code needed to access the various NXT input / output devices. The combined code is then automatically cross-compiled and linked into an executable which is downloaded to the target and run.

As shown in Fig. 2, it is also possible to include code for 'External Mode' communications which enable the user to interact with the NXT target at runtime, (as indicated by the dotted line in the figure). This feature is very useful for tuning model parameters during execution and for monitoring data, but is not yet implemented in the VU-LRT toolbox. A variety of third party, public domain tools are required to perform the cross-compilation and download operations outlined above, and the executable runs under a real time OSEK operating system that has been developed for the NXT [21]. The installation of these tools can be complex, so the VU-LRT toolbox ships with an automatic installer for the entire tool stack. The VU-LRT toolbox (and installer) are currently available for download from the MathWorks fileshare site [22].

From a user-perspective, the build and compilation details shown in Fig. 2 are largely transparent, and target specific features are encapsulated within the user-friendly VU-LRT blockset shown in Fig. 3. The user can drag and drop any of these blocks into their design in order to access any of the NXT's in-built features or attached sensors and actuators. A summary of these blocks and their function is as follows:

- Battery Volts – outputs the current battery voltage
- Time – outputs the time in ms since the model execution began
- Run Button – outputs a 1 if the 'Run' button is pressed, else 0
- Enter Button – outputs a 1 if the 'Enter' button is pressed, else 0
- Light Sensor – outputs a measure of the light received by the light sensor

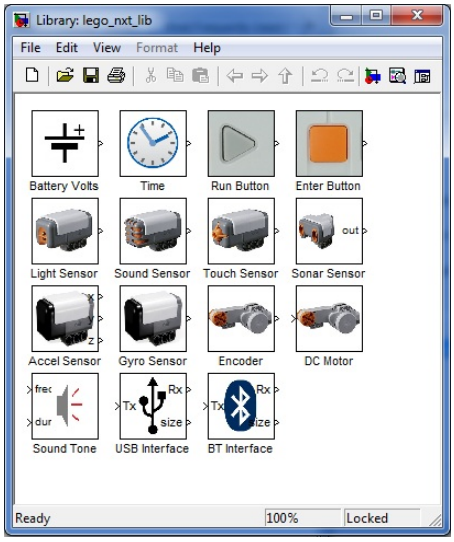

Figure 3. The VU-LRT Blockset

- Sound Sensor – outputs a measure of the sound intensity received by the sound sensor
- Touch Sensor – outputs a 1 if pressed, else 0
- Sonar Sensor – outputs the distance to the closest object in view by the sonar sensor
- Acceleration Sensor – Outputs acceleration data in three axes (x, y and z)
- Gyro Sensor – outputs the rotation rate of the sensor
- Encoder – outputs the number of encoder pulses received as the motor rotates
- DC Motor – sets the applied motor voltage as a percentage of battery volts
- Sound Tone – sets the frequency and duration of tones driving the internal loudspeaker
- USB Interface – enables the model to communicate with a host PC over USB
- BT Interface – enables the model to communicate with a host PC using Bluetooth

Note that the USB and BT interface blocks provide a means for transmitting and receiving data between the host PC and the NXT during runtime. This is a very useful feature for communicating specific data values or vectors between host and target, although it is not as flexible as the external mode feature described above.

4. Tutorial guides and examples

The user manual for the VU-LRT toolbox includes a ‘getting started’ tutorial guide and set of examples. The first example, illustrated in Fig.4, is intended to verify the correct functioning of all the tools and the successful transformation of a simple model into a real-time executable. The model therefore uses only the built-in button-press and loudspeaker functionality of the NXT, and is designed so that the loudspeaker emits a tone whenever the button is pressed. When the user initiates the build process, the model is automatically transformed into a C program which is seamlessly cross-compiled and downloaded to the target over a USB cable. The downloaded executable is then selected using the brick’s menu/button interface, and executed. Correct execution of the model is easily verified by confirming that a tone is sounded whenever the ‘Enter’ button is pressed.

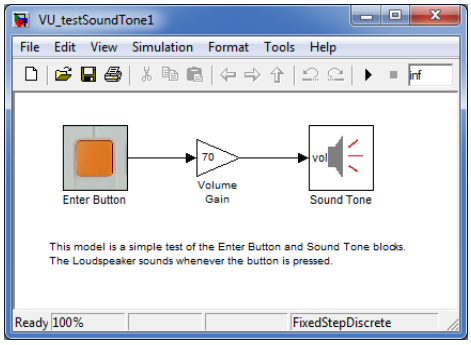

Figure 4. Example 1: Sound tone on button press

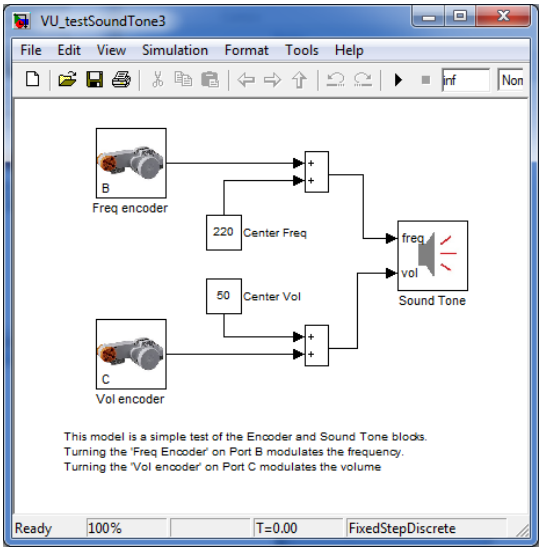

Figure 5. Example 2: Loudspeaker volume and amplitude modulation

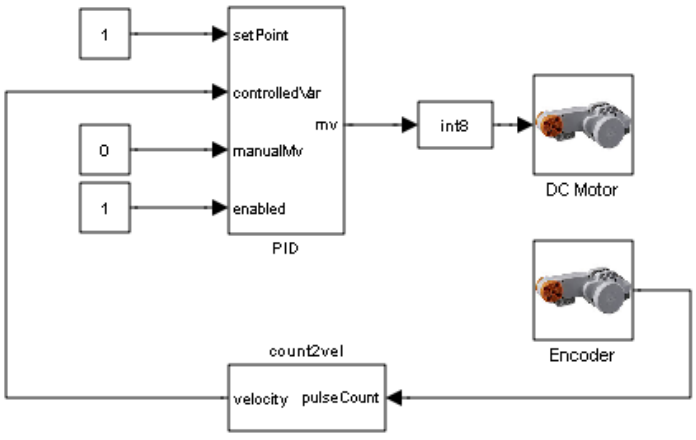

Figure 6. Example 3: A closed-loop dc motor speed control application

The second example, illustrated in Fig. 5, again uses the built-in loudspeaker, but uses sensor inputs from the integrated LEGO motor / rotary encoders to modulate the loudspeaker volume and frequency about some constant pre-specified values. When executing in real time, the user can therefore adjust the emitted tone amplitude and frequency by turning a small wheel or dial attached to each motor/encoder device.

A third example combines sensing and actuation in a simple feedback control loop. The example is based on a simple speed control requirement which might form part of a larger robotics system design. Motor speed is a function not only of the applied voltage, but also of the load. Open loop speed control, based on applied motor voltage, will therefore lead to speed variations when load disturbances, such as the robot needing to climb a hill, are encountered. However, a constant speed can be maintained using encoder feedback to detect differences between the actual speed and the setpoint, and passing this error through a standard Proportional, Integral, Derivative (PID) controller. Such an algorithm cannot be implemented effectively using the 'remote control' approach because of the (time-varying) delays in the communication channel, and the tight timing requirements needed for accurate speed estimation. However, the closed loop control algorithm is readily implemented using the VU-LRT blockset as illustrated in Fig. 6. The controller proportional, integral and derivative gains are set by means of the dialog window that is summoned by double-clicking on the PID block. After building and downloading the model executable to the target, students can then see for themselves how the motor torque increases so as to maintain speed when loads are applied. They can also repeat the experiment with different controller gains in order to observe how this affects the resultant closed-loop behavior.

5. Conclusion

Advances in low-cost target hardware, and automatic code generating software tools provide an opportunity for educators to engage students in challenging embedded system applications without the traditional difficulties associated with low-level hardware programming. In particular, the Villanova University LEGO Real Time target (VU-LRT) enables users to access the capabilities of the LEGO NXT brick (a powerful 32-bit machine), directly from a high-level Simulink environment. The motivation and design decisions underlying the development of the toolbox have been discussed, together with some details of its architecture. From a user's perspective, the VU-LRT blockset provides a strong base set of input/output blocks which can be used like any of the other standard Simulink blocks in student designs. Several simple examples drawn from the VU-LRT 'getting started' guide have been presented. The examples demonstrate the ease with which user designs can be implemented in real time hardware on the NXT, and the broad range of potential applicability to students at many different levels – from freshman thru to graduate students.

The toolbox has only recently been developed, and will be used for the first time in Spring 2011 in a senior-level Machine Learning course. Students will implement a genetic algorithm to enable a spider-like robot to learn how to walk. It is also intended to use the toolbox in a senior-level Automatic Controls course where the students will stabilize a Segway-like mobile inverted pendulum. Further work will be required to assess the learning benefits of these curricula

innovations and to develop additional laboratory modules which use the toolbox to enhance student learning. Further toolbox development is also required to add 'external mode' capability for host-target communication, although Bluetooth and USB communications are already in place.

Acknowledgements

The authors gratefully acknowledge support for this project from the National Science Foundation (DUE No. 0837637 [23]), and the MathWorks Inc. This work is neither endorsed nor maintained by the LEGO group. MindStorms, NXT, Technic and LEGO are trademarks of the LEGO group. Please direct all enquiries to the authors.

Bibliography

1. L. Greenwald, and D. Artz, "Teaching artificial intelligence with low cost robots," In Accessible hands-on artificial intelligence and robotics education, ed. L. Greenwald, Z. Dodds, A. Howard, S. Tejada, and J. Weinberg, pp. 35-41. Technical Report SS-04-01. Menlo Park, CA: AAAI Press, (2004).
2. S. Coradeschi and J. Malec "How to make a challenging AI course enjoyable using the RoboCup soccer simulation system, in RoboCup-98: Robot soccer world cup II: Lecture notes in artificial intelligence, vol. 1604, pp.120-124, ed. M. Asada and H. Kitano. Berlin: Springer, (1999).
3. M. Goldweber, et al. "The use of robots in the undergraduate curriculum: Experience reports," Panel at 32nd SIGCSE Technical Symposium on Computer Science Education, Charlotte, North Carolina..
4. F. Klassner, "Robotics as a Unifying Theme for Computing Curriculum 2001", National Science Foundation DUE #0088884, (2001).
5. D. Miller, "Walking Before Running: Filling the Freshman Engineering Gap by Building Mobile Stiquito(TM) Robots", National Science Foundation DUE #0088158 (2001).
6. N. Chao, "A Low Cost Hands-On Laboratory Experience for Introductory Engineering Students", National Science Foundation DUE #0125583 (2001).
7. L. Fairchild, "Robots in an Introductory Survey Course in Computer Science", National Science Foundation DUE #0087963 (2001).
8. N. McNulty, "Understanding Technology through Robots and Multimedia", National Science Foundation DUE #0088370 (2001).
9. Parallax Inc, website: http://www.parallax.com/
10. The Handy Board, website: http://www.handyboard.com/
11. Mobile Robots Inc, website: http://www.mobilerobots.com/
12. Arduino, website: http://www.arduino.cc/
13. LEGO MindStorms RCX, website: http://www.lego.com/
14. LEGO MindStorms NXT, website: http://www.lego.com/
15. Surveyor SRV-1, website, http://www.surveyor.com/SRV_info.html
16. Korebot II, website, http://www.k-team.com/mobile-robotics-products/korebot-ii
17. A. Behrens, et al. "MATLAB meets LEGO Mindstorms- A freshman introduction course into practical engineering", IEEE Trans. on Education, Vol.53, No.2, (2010), 306-317

18. T. Chikamasa, “Embedded coder robot NXT instruction manual”, www.mathworks.com/matlabcentral/fileexchange/13399/, 2009.

19. McNinch, L. C., Soltan, R. A., Muske, K. R., Ashrafiuon, H., Peyton-Jones, J. C. “An Experimental Mobile Robot Platform for Autonomous Systems Research and Education”, Proceedings of the 17th IASTED International Conference on Robotics and Applications, (2009): 412-418

20. McNinch, L. C., Soltan, R. A., Muske, K. R., Ashrafiuon, H., Peyton-Jones, J. C. “Application of a Coordinated Trajectory Planning and Real-time Obstacle Avoidance Algorithm”. Proceedings of the 2010 American Control Conference, June 30-Jul 2, Baltimore MD, (2010).

21. NXT OSEK/JSP, website, http://lejos-osek.sourceforge.net/

22. VU-LEGO Real Time Target, website, http://www.mathworks.com/matlabcentral/fileexchange/29857-vu-lego-real-time-target

23. J.C. Peyton Jones, F. Klassner, S. Kulkarni, C. Nataraj, “Introducing undergraduates to complex systems through rapid-prototyping of low-cost, networked mobile robots”. National Science Foundation DUE No. 0837637.