Virtual ARM Simulation Platform for Embedded System Developers

Alex Heunhe Han, Young-Ho Ahn, Ki-Seok Chung

Dept. of Electronics and Computer & Communications Engineering, Hanyang University
Tel : +82-2-2293-2123
E-mail: {alexhan, ghduddks84, kchung} @hanyang.ac.kr
Keywords : Virtual ARM, Simulation, Embedded System, Developing tools

1. A brief introduction

Virtual ARM Simulation Platform enables to observe the execution result of embedded software as if they are downloaded and executed on a real hardware ARM Platform. Developers can write program codes, build executable files, and verify their programs by using Virtual ARM Simulation Platform in the development host (PC). Since it doesn't need any hardware but PC, there is no downloading stage in developing procedure (See Figure 1). Major benefits that can be achieved by utilizing a Virtual ARM Simulation Platform are (1) reducing development cost, (2) lowering the entrance barrier for embedded system novices, and (3) making it easier to test and debug embedded software designs. This paper shows how Virtual ARM Simulation Platform is designed and how it can be used to reduce design time and cost.

2. Backgrounds

An embedded system is a special-purpose computer system designed to perform a small set of dedicated functions, sometimes with real-time computing constraints. It is usually embedded as part of a complete device which includes hardware and mechanical parts. Embedded systems span all aspects of modern life and examples of their use are numerous. Not only consumer electronics including personal digital assistants (PDAs), mp3 players, mobile phones but also telecommunications systems, transportation systems, and medical equipments employ numerous embedded systems such as mobile network system,

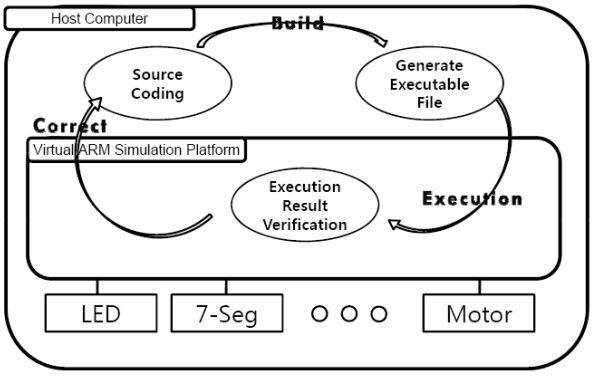

Figure 1. Development of Embedded System Software Using Virtual ARM Platform

anti-lock breaking system (ABS), GPS, electronic stethoscope. One of the most important hardware components of an embedded system is microprocessor. Microprocessor has played a main role in development of IT industry, especially in popularization of personal computer (PC) and internet. Each microprocessor has its own characteristics as it is used in various categories. A series of ARM processors have RISC-type architectures, and they have been widely used in a number of embedded designs. Because of not only their high performance and low cost but also power saving features, ARM variants are dominant in all corners of consumer electronics, from portable devices (PDAs, mobile phones, media players, handheld gaming units, and calculators) to computer peripherals (hard drives, desktop routers).

Virtual ARM Simulation Platform, which we are going to propose in this paper, is an ARM simulator designed considering target hardware. In contrast to

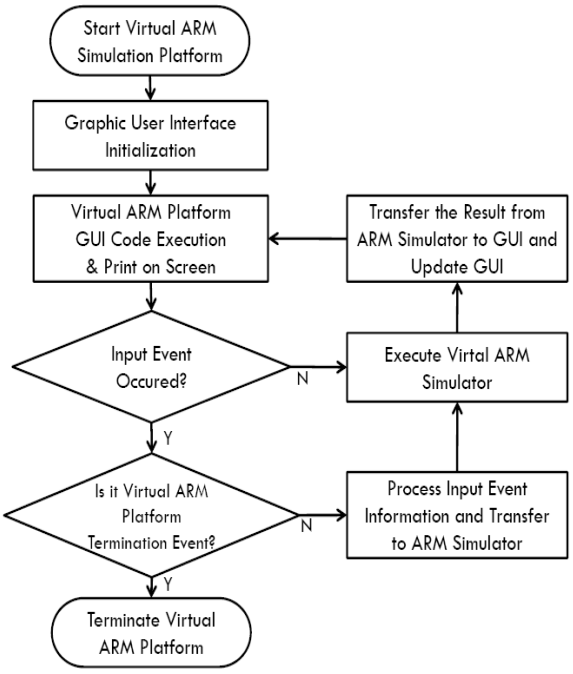

Figure 2. Execution Procedure of Virtual ARM Simulation Platform

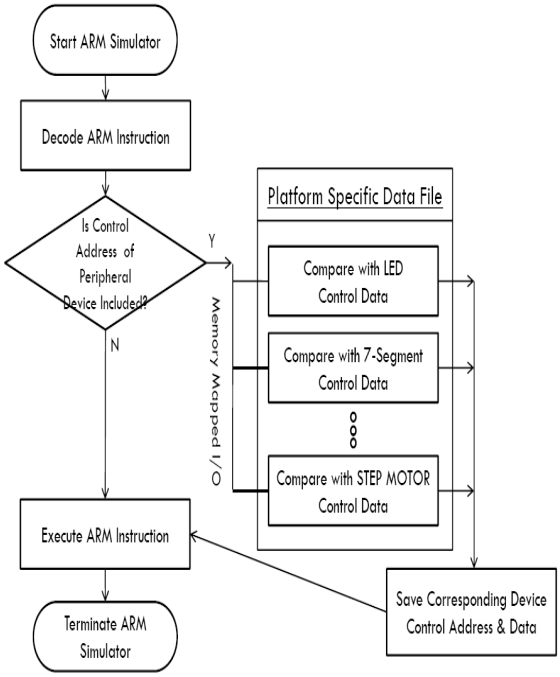

Figure 3. Execution Procedure of Virtual ARM Simulator

other virtual ARM machines that have been designed only in software model, our Virtual ARM Simulation Platform enables most similar operations to target embedded system. By "similar operations", we mean that our Virtual ARM Simulation Platform allows embedded system developers to develop and test their embedded application as if they do on a real H/W platform. To implement a target-specific Virtual ARM Simulation Platform, we have chosen SYS-Lab5000 A RM hardware platform, (designed by Libertron, Inc.,) as our target H/W platform. Since a target-specific virtual platform allows the target platform specific details to be tested without the real target platform, design time and cost can be greatly reduced. Especially, for educational institutions where a sufficient number of embedded equipments are not deployed, this type of target specific virtual platform will be greatly helpful to teach students how to design a target specific embedded system while minimally requiring the time to use the real hardware. Our Virtual ARM Simulation Platform implementation is based on ARM simulations by using SimIt-ARM simulator [1] and Graphic User Interface by using QT library [2]. Also to extend the target specific capability with ease, we have designed an emulator which automatically produces plat-form-specific environment settings from the given platform-specific information for easy maintenance and adjustment. Furthermore, we have implemented a timer interrupt handler for simple O.S (uC-OSII[3]) simulation.

3. Implementation of Virtual ARM Simulation Platform

We implemented Virtual ARM Simulation Platform offering extended control for peripheral devices which conventional ARM simulators cannot offer. By adding control codes for peripheral devices to existing ARM simulators such as SimIT-ARM or SimpleScalar, Virtual ARM Simulation Platform can control peripheral devices such as LED, 7-segment, step motor, etc.

Virtual ARM Simulation Platform consists of Virtual ARM Simulator, Graphic User Interface, Input Event Handler, Timer, and I/O Device Models. As Figure 2 shows, when Virtual ARM Simulation Platform is started, it initializes Graphic User Interface environment and displays the main GUI window on screen. Then Virtual ARM Simulator is executed, processing the instructions in the executable file and sending the results to Virtual ARM Simulation Platform. When Virtual ARM Simulation Platform gets

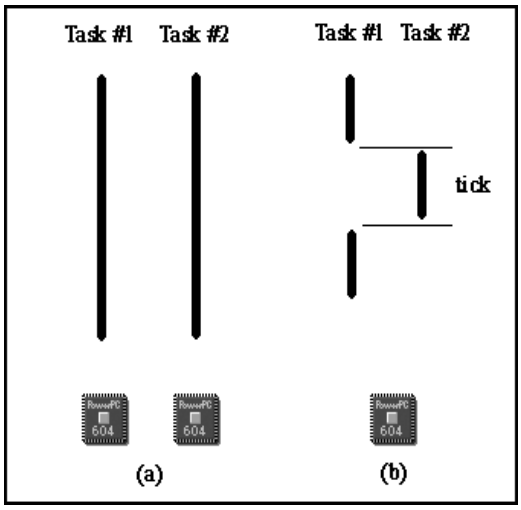

Figure 4. Parallelism vs. Concurrency

the simulation results from Virtual ARM Simulator, it updates its GUI in cooperation with the execution report from Virtual ARM Simulator. When Input Event Handler detects any input events, it analyzes the event and terminates the program if the event is the termination event. If the event is not the termination event, the handler sends the processed input event to Virtual ARM Simulator, which simulates the instructions with the transferred input data.

Virtual ARM Simulator not only decodes and executes the given ARM instructions just as a real ARM processor does, but also handles the transferred events from Virtual ARM Platform. As Figure 3 shows, when Virtual ARM Simulator decodes an ARM instruction, it checks whether the decoded instruction includes control address of peripheral devices. If it doesn't, Virtual ARM Simulator simply executes the ARM instruction and sends the results to Virtual ARM Platform. However, if it does, Virtual ARM Simulator has to do additional work. Since Virtual ARM Platform employs memory mapped I/O in controlling peripheral devices, a platform-specific data file contains device control register access addresses and corresponding device control addresses. By looking up H/W platform specific data, Virtual ARM Simulator finds which device it should control, and then finds how to control the corresponding devices. Finally, Virtual ARM Simulator processes the device control by sending the saved device control address and data to Virtual ARM Platform.

Also, by using timer implemented in Virtual ARM Simulation Platform, we enabled watchdog timer interrupt, context switching and simple OS simulation such as μC/OS-II. Figure 4.(b) shows the multitasking of μC/OS-II using timer of Virtual ARM Simulation Platform. Single processor can do multiple tasks concurrently by doing context switch for each preset time (tick) .

4. Experimental Results

We have carried out experiments to test the performance of our proposed Virtual ARM Simulation Platform. By porting uC-OSII [3] and running several tasks on Virtual ARM Simulation Platform (see Figure 5), we could see both the target real H/W platform and our Virtual ARM Simulation Platform act exactly same, as Figure 6.1 and Figure 6.2 show the identical execution results.

Task 1.
-Rotate STEP MOTOR
-Turn on LED LAMPS

Task 2.
-Count from 1 to 10 through 7-Segment
-Count from 1 to 10 through Dot-Matrix
-Turn off LED LAMPS

***uC-OSII controls task 1 and task2 using timer**

Figure 5. Multitasking Experiment using uC-OSII

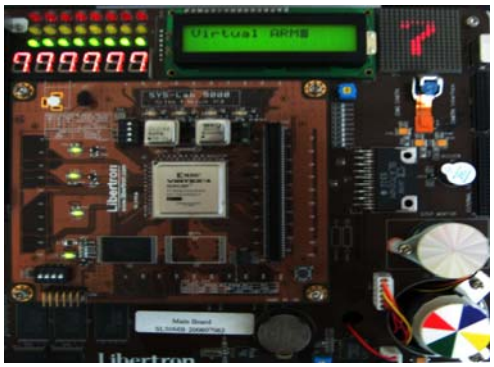

Figure 6.1. uC-OSII on the real board called SYSLab-5000

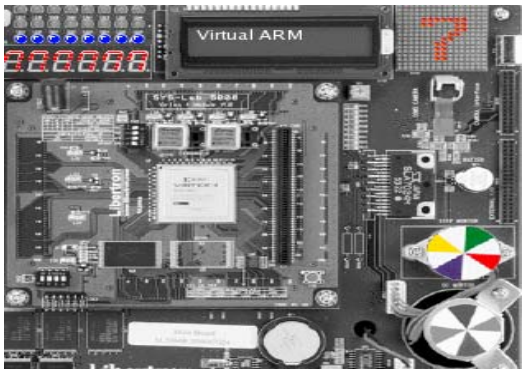

Figure 6.2. uC-OSII on Virtual ARM Simulation Platform

5. Conclusion

In this paper, we have proposed Virtual ARM Simulation Platform that offers developers an excellent test environment which is very similar to a real H/W ARM platform. By using our proposed Virtual ARM Simulation Platform, developers will be able to develop ARM based embedded system software without buying a target H/W platform. Since developers can test their program immediately in host PC without downloading executable files on the real platform, the overall development procedure will be faster and more comfortable. Also, a GUI interface which resembles the target H/W ARM platform is user-friendly, and therefore, will lower the entrance barrier for embedded system novices. Especially, Virtual ARM Simulation Platform will be ideal as a college-level education tool for the students who need to practice their knowledge in embedded system software without any time, spatial and financial limits.

Acknowledgement

This research was supported by the MKE(Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute for Information Technology Advancement)"
(IITA-2008-C1090-0801-0045)

References

[1] http://simit-arm.sourceforge.net

[2] http://trolltech.com/products/qt

[3] Jean J. Labrosse, MicroC/OS-II Real Time Kernel 2/E, R&D Technical Books, 2002.

[4] Gon Kim, Sang-Young Cho, and Jungbae Lee, “Virtual Prototyping Environment on ARMulator”, *Korea Computer Congress 2004 Vol. 2.*, 2004, pp. 592-594.

[5] P. Schaumont, D. Ching, I. Verbauwhede, "An interactive codesign environment for domain-specific coprocessors," ACM Transactions on Design Automation for Embedded Systems, January 2006.

[6] W. Qin, S. Malik. Flexible and Formal Modeling of Microprocessors with Application to Retargetable Simulation, Proceedings of 2003 Design Automation and Test in Europe Conference (DATE 03), Mar, 2003, pp.556-561.

[7] W. Qin, S. Rajagopalan, S. Malik, A Formal Concurrency Model Based Architecture Description Language for Synthesis of Software Development Tools, ACM 2004 Conference on Languages, Compilers, and Tools for Embedded Systems, June 2004, pp. 47-56.

[8] Rajesh Kumar Gupta, "Co-synthesis of Hardware and Software for Digital Embedded systems," PhD thesis, Stanford University, April 1991.