Virtual Development Environment for Smart Card Applications

Sang-Young Cho
Computer Science and Engineering Department
Hankuk University of Foreign Studies
San89, Wangsan, Mohyeon, Cheoin, Yongin, Kyeonggi
REPUBLIC OF KOREA
sycho@hufs.ac.kr http://cse.hufs.ac.kr

Abstract: - Virtual development environment increases efficiency of embedded system development because it enables developers to develop, execute, and verify an embedded system without real target hardware. This paper deals with an implementation of a virtual development environment for Smart Card applications. Usually application developments are performed with a evaluation kit that equipped with a hardware system, software development tools, and optionally emulator for debugging. The hardware-based development kit has some demerits in maintenance. The environment is developed based on ARMs ARMulator that is an instruction set simulation environment. By adding hardware IP modules such as Memory modules, UART, Interrupt controller, Timer, Watchdog Timer, and additional host applications to emulate a host interface, the ARMulator environment is expanded to a virtual software development environment for basic smart card applications. NOR flash memory models and SRAM models are implemented in our environment. In addition, a chip operating system is implemented to construct a DCAS development environment.

Key-Words: - Smart card, Simulator, Virtual environment, DCAS, ARMulator, Modeling

1 Introduction

A smart card, chip card, or integrated circuit card (ICC), is any pocket-sized card with embedded integrated circuits which can process data. This implies that it can receive input which is processed — by way of the ICC applications — and delivered as an output. There are two broad categories of ICCs. Memory cards contain only non-volatile memory storage components, and perhaps some specific security logic. Microprocessor cards contain volatile memory and microprocessor components [1]. Recently powerful smart cards are shown from various smart card manufacturers to support various high-end smart card applications such as mobile phones, Java cards, electronic passports, banking cards, transportation cards and e-commerce applications [2]. Usually application developments are performed with an evaluation kit that equipped with a hardware system, software development tools, and optionally emulator for debugging.

There are some demerits to use a real hardware for the development of smart card applications. First, the even well-shielded hardware kits are vulnerable to be damaged by frequent programming and program flash memories. Second, the debugging has some constrains to view internal states and detailed behavior of applications. Third, the programming and debugging process is somewhat long because the process is related with the hardware modification. Forth, the price of evaluation kit and development tools is usually expensive. Therefore it is difficult for team members to develop applications concurrently.

Traditionally software simulators are used to replace hardware kits for the purpose of overcoming the demerits in embedded system software development demerits. Also similar approaches are adapted to smart card software development [3]. In this paper we focus on the development of software simulator for smart card application development and describe a chip operating system dedicated to Samsung’s smart card system.

Especially this paper deals with a simulation-based software development kit for DCAS[4]-related smart card applications. Actually, the kit is constructed on the level of virtual development environment (VDE). A VDE is virtually built inside a computer, and it provides the similar appearance as real hardware and simulates various peripherals attached outside of the board and functions of them as well. VDE is used to verify a hardware prototype, develop software without the real hardware, or be used for co-design of hardware and software for embedded systems. This environment usually provides a hardware simulation model, simulation engine, and other tools that are useful in software development. Thus it increases the efficiency of the embedded system development [5-7]. Though it does not provide with the same appearance of the target system, application software could be developed and fully verified using hardware or software IP models.

Virtual Platform [5] is Synopsys’s commercial virtual development environment. It supports many
different types of processors such as ARM, X-Scale, and MIPS that are usually used in embedded systems. Virtual Platform supports to develop software for target systems. RealView Soc Designer [6] is comprehensive SystemC-based SoC development environment provided by ARM. It consists of fast and easy modeling and simulation, and debugging tools. Additionally, it enables a system or hardware developer to compose the most suitable architecture quickly and accurately, and software developers to develop software before the actual hardware comes out by providing VDE. Visual Elite [7], made by Mentor Graphics, is a set of ‘Fast ISS’ models and ‘platform-based packages’, which are composed of TLM-based busses, memory, and peripheral device models. According to its setting, it can comprise and run many different abstract-level systems, and debug-related software and firmware. Also, it can create a virtual environment for fast software development. The commercial VDE’s described above are tightly integrating the hardware simulation tools and software development tools. Therefore, it limits the flexible use of various software tools and hardware simulation tools from different companies or organizations. Also, the VDE’s are too expensive to be used in team projects.

In this paper, we describe the design and implementation of a virtual development environment for developing smart card application systems that runs on systems based on ARM cores. We construct the kit based on ARMulator that is an ARM core instruction set simulator of ARM cooperation. The ARMulator environment uses a specific hardware modeling method. We modeled the AMBA bus that is an in-chip bus scheme developed by ARM and we extended the environment by attaching several hardware IP modules such as Memory modules, Memory Protection Unit, Interrupt controller, UART, Timer, and Watchdog Timer. NOR flash memory models and SRAM models are implemented in our environment. In addition, a chip operating system is implemented to construct a DCAS development environment. The kit is constructed with ARM7TDMI core that has the same functionality with S100 that is usually used for smart cards.

The rest of the paper is organized as follows. In section 2, we describe the ARMulator environment that is a basic platform of our kit. In section 3, we explain the design and implementation of the kit for smart card applications. Finally, Section 4 concludes the paper with final remarks.

2 ARMulator Environment

For the processor vehicle of a virtual software development environment for smart card applications, ARM7TDI core is selected due to its popularity in high-performance smart card products. ARM processor cores are occupying more than 75% market-share around the world in 32-bit embedded RISC microprocessor market and the most of mobile terminal equipments are adopting ARM cores as their main control processors. We also choose the ARMulator environment as the base one of our kit because it is too time-consuming to build our own ARM processor core simulators and to manage to follow up the rapidly-announced ARM core series. Furthermore, most of software developers for ARM systems have an ARM’s software development environment such as ADS (ARM Developer Suite) and RVDS (RealView Developer Suite) series and ARMulator is one of components of the ADS or RVDS series [8-9]. An ARMulator-based virtual software development environment may enable developers to have a virtual environment with a minimum cost.

ARMulator is a virtual board model that stands on the basis of cycle-based instruction set simulator of ARM processor cores. As it has a simple memory model and a standard hardware IP, the performance of hardware and software can be estimated before the actual implementation of the product by using the software’s profiling function even if no tangible board exists. Figure 1 shows the ARMulator framework.

![Figure 1. ARMulator DLL structure](image)

ARMulator consists of simulation kernel and extension modules. Simulation kernel is in charge of core simulation and external interface with RDI debuggers or extension modules. Extension modules include basic peripherals such as timer, watchdog timer, time tick, and interrupt controller. Even though software development is possible with those peripherals as well as core simulator in the ARMulator virtual environment, it is far from a real environment where various peripherals are working. ARMulator is open to expand peripherals with the extension interface of memory system. Fig.2 shows
3 Design and Implementation

3.1 Extension of ARMulator environment

ARMulator can be extended by adding any hardware that is modeled in C language as DLL file. Fig. 3 shows the target system hardware modules to be added to form a virtual development environment.

The target system modules are designed and implemented by a simple smart card model. Usually a smart card suffers from very constrained resources. The card has ROM for an in-chip operating system, SRAM for data area, and Flash memory for application-specific codes. We implemented the all the memory modules using ARMulator memory with operation restrictions. The sizes of ROM, SRAM, and Flash memory are assumed to be 32 Kbytes, 50 Kbytes, and 500 Kbytes, respectively by referencing to Samsung’s smart card specification [10]. The ARM7TDMI IIS simulator of ARMulator is used as the CPU core of the target system. The ARM7TDMI has the same functionality with SC100 that is a secure core of ARM company. UART is for communication interface with external card readers. The interface complies to the ISO 7816 protocol. Two 16-bit timers and one 20-bit watch-dog timer are implemented. Interrupt controller supports several interrupt sources from internal HW IP modules. Currently, the memory protection unit and crypto engine are not implemented. Also we ignore clock and reset control or behaviors. We assume the clock is fixed to an external clock that is 3.57 MHz.

Even with this simple smart card simulation environment, we can develop almost all main parts of smart applications because the other parts can be simulated with software easily. The image for the smart card is developed with ADS 1.2 IDE[8] and the image is downloaded and debugged with AxD debugger of ADS 1.2. A host system interacts with a smart card by connecting them with a smart card reader. A host program manages smart cards with APDU commands.

3.2 Host emulation program

To verify the behavior of smart card application programs, there needs a program to interact with smart cards for sending and receiving APDU commands. We make a program that can send commands and receive responses to/from the implemented smart card simulator. The program is customized to be used in DCAS application developments. So, the program has functions of detecting simulator, connecting simulator, general APDU processing, and DCAS-specific operating such as client downloading, deleting, starting, ending, and reading log information. Fig. 4 shows the host emulation program.

The emulation program is connected with the UART module of the implemented simulator using IPC of Windows systems. The right side of the program shows the APDU command processing status and displays received data from a smart card. The log data logged in a smart card can be read with “Read Log Data” command. The emulation program can be used when a real smart card is connected with the host systems. Fig 5 shows the overall application development environment for smart card application developments.
3.3 Chip operating system for DCAS application

DCAS (Downloadable Conditional Access System) is a proposal advanced by CableLabs for secure software download of a specific Conditional Access client (computer program) which controls digital rights management (DRM) into an OCAP-compliant host consumer media device [12]. Fig. 6 shows the DCAS operation assumed in this paper.

To develop DCAS applications, a smart card should handle several clients and thus needs a chip operating system (COS). The main role of clients is key management. DCAS Monitor performs logical function of the overall Bootloader and management of threads for clients and communication. SM Bootloader initializes the smart card hardware and manages cryptographic, client, and memory. We designed and implemented simple COS. The COS supports threads for DCAS monitor and clients and communication facilities for communication among a host program, DCAS monitor, and clients. We assume that maximum three clients of legacy CAS, DRM, and ASD types are supported and each client can be independently developed regardless of underlying COS implementation. This means that each client can be relocatable at downloading or run times. The COS supports generic asynchronous Send and Receive communication APIs. Each thread can occupy CPU by three context switching types. After smart card hardware initialization, the initial thread, DCASMon, occupies the CPU by initial context switching. The initial thread turns on each thread for a client and makes the thread run by calling software context switching function. Timer-based interrupt makes ready threads share CPU by calling interrupt context switching function. The operation of the implemented COS is verified with a small-size application of three clients.

With the simulator for the smart card hardware, the host emulation program, and the COS, we construct a smart card software development environment based on the ARMulator simulation tools.

4 Conclusion

In this paper, we describe the design and implementation of a software development environment for smart card applications. We construct the simulation environment based on ARMulator that is an ARM core instruction set simulator of ARM cooperation. The simulation-based software development tool can overcome demerits of hardware-based software development.

The ARMulator environment uses a specific hardware modeling method. We modeled the AMBA bus that is an in-chip bus scheme developed by ARM and we extended the environment by attaching several hardware IP modules such as Memory modules, Memory Protection Unit, Interrupt controller, UART, Timer, and Watchdog Timer. NOR flash memory models and SRAM models are implemented in our environment. In addition, a chip operating system is implemented to construct a DCAS development environment. The kit is constructed with ARM7TDMI core that has the same functionality with S100 that is usually used for smart cards. A host emulation program is developed to verify the smart card application program behavior.

With the developed environment, various smart card applications can be developed with low costs. The simulation environment is very flexible because that is based on the popular software development tool such as ADS or RVDS series that supports various CPU types.

For further works, more hardware IP modules related to smart card hardware should be supplemented. In current implementation, MPU and Crypto engine are not implemented and reset and clock operations are assumed to be fixed. Also each IP modules have room to improve functionality.
Acknowledgements: This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) Support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2010-C1090-1031-0004)

References: