DOI:10.5392/IJoC.2011.7.1.023

Analysis of Verification Methodologies Based on a SoC Platform Design

Je-Hoon Lee
Div. of Electronics and Information Communication Eng.
Kangwon National University, Samcheock, Gangwon 245-711, Rep. of Korea

Sang-Choon Kim
Div. of Electronics and Information Communication Eng.
Kangwon National University, Samcheock, Gangwon 245-711, Rep. of Korea

ABSTRACT

In a SoC (system-on-chip) design, a design complexity is a big bottleneck. In order to overcome the design complexity, platform based design method is widely adopted for designers. Most complex SoCs need a heterogeneous design development environment for hardware and software co-design. In this paper, we discuss about some kinds of verification approaches with platform based design methodology at various abstraction levels of SoC design. We separate the verification process to two steps according to the different levels of verification. We employ a flexible SoC design environment to support simultaneous hardware and software development. We demonstrate the verification strategy of a target SoC design, IEEE 802.11a WLAN SoC.

Keywords*: SoC design, Platform-based design, Hardware-Software Co-design, Verification.*

1. INTRODUCTIONS

SoC solution provides big advantages and some difficulties for designers. For example, benefits of using SoC solution are size reduction, low cost, lower power consumption and increased performance. However beyond these advantages, design complexity is drastically increased because technological advancements let us integrate a lot of functions into a single chip.

As the complexity of SoC design is constantly growing and reusable IP (intellectual property) libraries are becoming wealthy, the main SoC design issue shifts to the verification method to handle the complex SoC system easily. There is a research result that showed the verification typically consumes about 70% of the design effort [1]. In particular, the recent SoC designs employ heterogeneous development environments to support the simultaneous development of hardware and software designs. It also should accommodate the flexibility of SoC design owing to the frequent system expansion. Thus, a seamless hardware and software co-verification environment is becoming important to verify the complex SoC design fast and accurate.

Recently, there are some hardware and software co-verification researches [2-5]. J. Bieger presented a rapid prototype method for configurable SoC platforms based on simulation approach. T. Li introduced the software and hardware co-verification environment, SoC-Gen, and gives a detailed explanation on the subsystem, SoC-CBSHVE. It does not support the link between the architecture verification and lower levels of validation. Recently, a low-cost co-verification solution consisting of a hardware emulator based on FPGA and an embedded processor is introduced [6]. It provides a good visibility for the internal signals of the system design mapped in the emulator. It is useful to design the complex SoC design because of its strong emulation function. However, the verification of a complex SoC design is achieved by the several validation levels such as system architecture, stand-alone IP, subsystem platforms, SoC integration, and system/chipset [7]. Thus, the linking between the architecture verification and lower levels of validation is required.

In this paper, we present analysis of some kinds of verification approaches based on SoC platform by experimental design of IEEE 802.11a WLAN implementation and discuss about the performance and ease of verification. The baseband processor is implemented by hardware design and the MAC is implemented by software design. This paper is organized as follows. Design verification issues and approaches are explained in section 2. Section 3 introduces case study by design verification of IEEE 802.11a. In section 4, simulation results are presented. Finally section 5 gives conclusions.

* *Corresponding author. E-mail : sckim@kangwon.ac.kr*
Manuscript received Oct.23, 2010 ; accepted Mar.16, 2011

2. VERIFICATION STRATEGY

The designers deploy verification environment utilizing C/C++ and SystemC languages through clearly predefined verification strategy and simulate a complex SoC with its embedded software. Currently we use verification strategies using on platform based co-simulation for high-level abstraction and co-verification for high-speed hardware and software emulations. This strategy supports the concurrent hardware and software co-development as shown in Fig. 1. A co-simulation environment employs an ISS (instruction set simulator) that models the target processor and a SystemC/HDL simulator based on a FPGA. A co-verification environment employs a high-speed FPGA-based hardware emulator and an actual embedded processor. It is used to verify the whole SoC integration before the fabrication of a target SoC.

The co-simulation environment is only a software platform. Thus, it is easy to use and convenient to debug in design verification for each IP and functional level system verification. However, it spends too much time to verify the entire system in cycle or transaction level as the complexity of a SoC grows. Thus, the co-verification environment is required to increase the simulation speed by employing actual processor instead of ISS and a hardware simulation accelerator. The hardware and software designs, API, and testbenches can be shared for both environments. The major benefit of the proposed verification technique is its straightforward verification flow between different design steps.

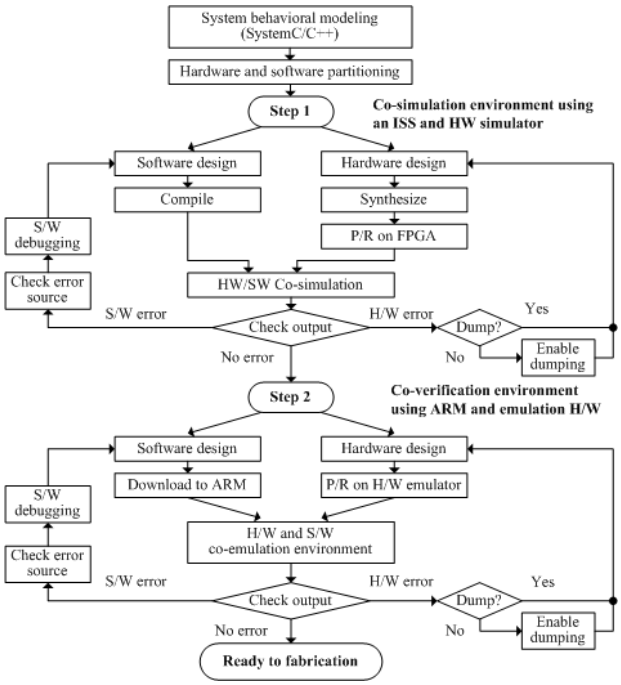

Fig. 1. Design flow for heterogeneous SoC design.

2.1 Co-simulation for high-level abstraction

Design effort for complex SoC is significantly decreased by adopting platform based co-simulation environment into SoC design domain. The co-simulation environment might contain a predetermined embedded processor, operating system, bus architecture, and various intellectual properties. By adding its embedded software or custom logic into this co-simulation environment, the designers can quickly produce a chip without redesigning the whole thing from scratch.

The basic approach is making virtual SoC platform environment through prototyping or emulation based approach. In virtual prototyping, the whole environment is made by specific kind of software model. Although virtual prototyping has high flexibility and high accuracy, verification performance is significantly affected by third party co-simulation environments. In contrast, emulation is physical model of SoC platform so the performance could be high but it is significantly affected by communication interface between reconfigurable, programmable devices and accelerator engines used to create virtual model of a co-simulation environment.

In this paper, we use a co-simulation environment mainly consisting of two parts; an ISS and a hardware simulator. An ISS is used to execute software design of target SoC. A hardware simulator is to implement the hardware part using SystemC and HDL. A transactor is used for the communication between them. Using this environment, co-emulation is possible, where source-level debugging through C/C++ debugger and waveform-based hardware debugging can probe the internal node and register value to debug target SoC as easy as a conventional HDL simulator. However, a simulation for hardware part spends too much time as the complexity grows. Thus, we simply verified both stand-alone IPs and the system-level functionality in this phase.

2.2 Co-verification based on high-speed H/W emulation and embedded processor

After system-level validation of the functionality using a co-simulation environment, we employ a co-verification framework to validate the whole SoC system integrated in single hardware platform, which contains an embedded processor core and a fast hardware emulator employing a hardware accelerator to reduce the verification time.

In instance of co-design where a processor and custom logic are both a part of a design, the interface between hardware and software becomes an area of increased focus and attention. Validating of hardware and software function correctness together can become an important aspect in the overall verification process. Therefore, efficient HW/SW co-verification methodology is necessary and it can help uncover a range of HW/SW interface problems. Co-verification has some advantages compared with respective verification. Key concept behind co-verification is to merge the respective debug environments used by hardware and software teams into a single framework. It provides designers an early access to both hardware and software components of the designs. Consequently, it reduce overall project cycle time [8].

Figure 2 shows three basic cases of verification models with abstract model. In first case as shown in Fig. 2a, S/W design might be designed by pure C/C++ languages and H/W design might be accelerated by hardware accelerator such as FPGA engine. In such case, verification performance depends on bandwidth of communication interface and speed of H/W accelerator. In second case as shown in Fig. 2b, S/W design might be accelerated using ISS and H/W design same as previous case. The ISS provides some memory model and

additional peripheral modules that can be implemented as C/C++ abstract models and it can communicate with core through memory model. So a custom HW design becomes one of the peripheral modules. In such case, overall verification performance is significantly affected by memory model. Some kind of methods such as 'coherent memory server' used to accelerate performance. In third case as shown in Fig. 2c, HW and SW design are both accelerated using real hardware engines that are used to emulate virtual SoC platform. In such case, overall verification performance is significantly affected by communication interfaces and their bandwidth.

The verification of the hardware part is performed through an emulator and thereby simulation speed is very high and the verification coverage is enhanced. It is executed by checking an output of emulation and checking a waveform that is dumped during emulation. Checking an output of emulation is validation process that is performed by comparing an input and output data. The dumping is a process to sample the signals and to store them in the internal memory of the emulator. The contents of the memory are sent to the host computer after the emulation is completed, and they are translated into an electronic waveform which is used to present the cycle accurate debugging information of the hardware design.

The verification of software design is achieved by the processor core using In Circuit Emulator (ICE) via JTAG port. An ICE provides the strong source-level debugging by handling the processor core directly. In addition, it can work together with the hardware emulator based on FPGA. Thus, waveform-based hardware debugging through a hardware emulator that can probe the internal node and register value gives a strong debugging function for the complex SoC.

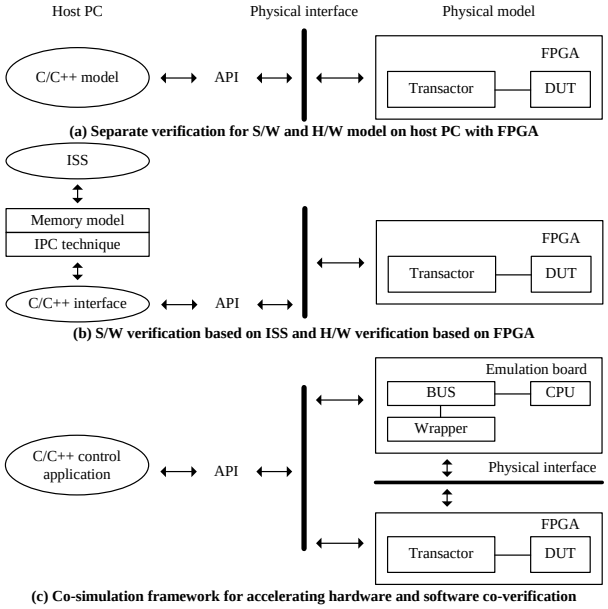

Fig. 2. Various verification models of co-design.

The verification approach based on co-verification platform gives us this possibility to move our attention on the functionality of the entire process of baseband processor to HW/SW interfaces between software implementation and hardware design thereby it reduces the possibility of a problem remaining hidden until system integration. The main role of our verification framework is to provide efficient environment to detect design bugs as well as possible.

3. CASE STUDY

We applied previously discussed verification approaches to implementation of IEEE 802.11a. The IEEE 802.11a specifies broadband communication systems using OFDM (orthogonal frequency division multiplex) in 5 GHz band with data rates ranging from 6 to 54 Mbit/s. It consists of two main parts including MAC and baseband processor (PHY) [9]. In fact, the general approach of implementing WLAN is MAC as embedded software and baseband processor as custom hardware design. We maintain this approach. The complete 802.11a compliant baseband processor was modeled in VHDL and synthesized with at RTL level. The 802.11a compliant MAC functions was implemented in C/C++ and compiled by Visual C6.0, ARM ADS 1.2.

Thus, pure C model of MAC runs on ISS model of ARM9, ARMulator and baseband processor mounted on *iPROVE* prototype board [6] as FPGA engine as shown in Fig. 3. An ARMulator provides an environment for the development of ARM targeted software on range of none ARM based host system. Also we modeled some extra peripheral models including virtual LCD display and interface of the FPGA card with ISS.

We choose a SoC platform including ARM core, AMBA bus architecture. Because today ARM core family and AMBA bus architecture is de facto architecture in SoC. The ISS and hardware simulator are connected with each other through PCI bus. Specific API (application programming interface) functions are to make transaction between C/C++ model and baseband processor. In order to handle this transaction level signal and cycle accurate design, a transactor is implemented with HDL and the transactor interprets the transactions and activates the pin signals.

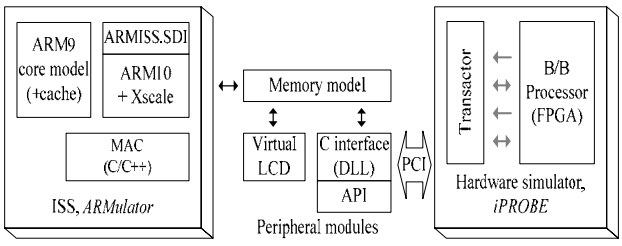

Fig. 3. Co-simulation framework for behavioral system model.

A basic architecture of the co-verification platform consists of three major parts as shown in Fig. 4; hardware emulator, *iPROVE* [6], ARM-based board, *PROBASE* [10], and interface units. The ARM-based board contains ARM core, FPGA, memories, and some peripherals. It is used to integrate the software part of target SoC and the interface module. The previously verified IEEE 802.11a compliant MAC compiled in an ISS is reused by porting as a firmware on ARM core. The

FPGA based emulator accommodates some parts of hardware in its FPGA. The baseband processor is synthesized and mapped into FPGA in a hardware emulator. The communication between an ARM-based board and a hardware emulator is achieved by DPP interface. We implemented application program on host PC that is used to generate testbenches to whole system in the platform while controlling overall verification process. A PCI interface is to communicate the testbench between this co-verification platform and the host PC. The bus splitter is used to connect with the FPGA based hardware emulator by making two different system buses to only one system bus. In our implementation, the AMBA bus is prototyped in FPGA of ARM based board. Design in this board and a FPGA based emulator are logically connected through AMBA bus splitter.

In here, some extra wrappers were modeled in VHDL which is a bridge to communicate DPP bus and AMBA bus as shown in Fig. 4. In order to handle all verification process we implemented control application on host pc and it gives commands through RS232, PCI to MAC and baseband processor respectively. Table 1 shows fundamental levels and corresponding operations, data types of IEEE 802.11 design. In application level, we use real time video data stream capturing from web camera for the verification. Thus MAC block makes MPDU (MAC protocol data unit) frame from video data stream and then it is divided into transmission sub blocks that are called PPDU (packet protocol data unit) and send it to baseband block. At interface level, it will be sent byte by byte between two units.

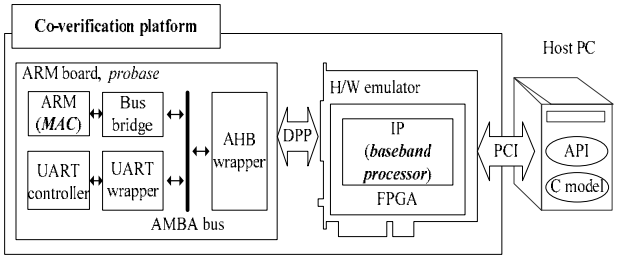

Fig. 4. Co-verification platform with IEEE 802.11a WLAN SoC.

Table 1. Transaction type of data.

Level	Operations	Fields
Interface	Send, receive	Byte
Unit	Assemble, error checking	PPDU
Final assembly schedule	Capacity control	MPDU
Application	Initiate, transmit, complete	Real-time video data

4. SIMULATION RESULTS

This section summarizes verification result. In a co-simulation environment composes of an ISS that models ARM core and 6 million gates FPGA. In a co-verification environment composes of ARM-based board as a SoC platform and a hardware emulator including million gates FPGA. A hardware emulator accommodates the baseband processor IP and some peripherals.

We have compared 5 kinds of verification approaches in term of performance. First approach is RTL level verification of baseband processor by software simulation. A second approach has two modes due to *iPROVE* FPGA [6] card which supports two kind of mode including cycle level mode based on cycle and transaction level mode based on transaction. We verify own design in both mode. In the third and fourth approaches, hardware part is same as previous. But software part that MAC design is accelerated using ISS and physical model of ARM processor. Thus in Fig 5, we present performance of each verification model. Second approach reaches the highest performance in transaction level mode. But in system level, performance is decreased because of co-verification environment. We assume that main reason of performance decreasing is due to the intercommunication between hardware and software design parts as shown in Fig. 6. For example, in ISS case, it uses memory sharing technique to exchange data between virtual ARM core module and peripheral modules.

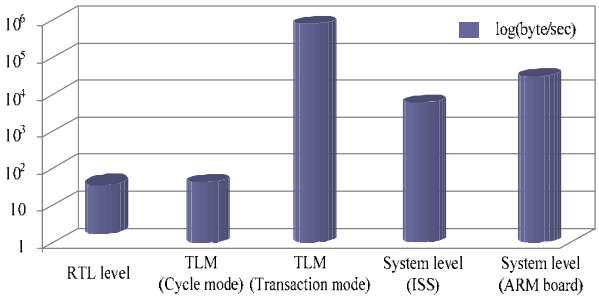

Fig. 5. Performance of verification systems.

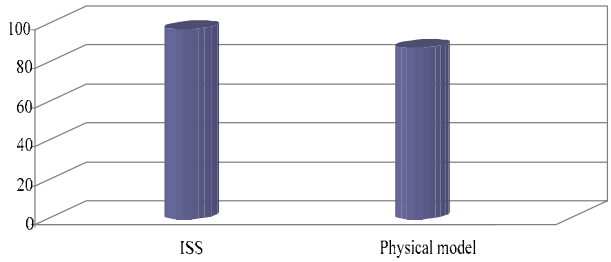

Fig. 6. HW/SW interface latency in overall simulation time.

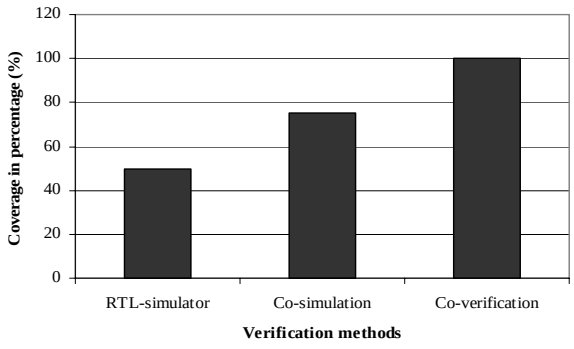

Fig. 7. Verification coverage of each verification method.

Figure 7 shows verification coverage of each verification

method. The RTL-simulator has around 50% of coverage. Verification coverage has following metrics such as function coverage, statement coverage, branch coverage and interface coverage. In the RTL-simulator, a simulation is very slow and we assume that this satisfies function and statement coverage. If we assume that each verification metrics has 25% of the full verification coverage, then RTL-simulator's verification coverage would be 50% of the full verification coverage. In the co-simulation environment, emulation has high speed so it is able to fulfill the function coverage, statement coverage and branch coverage. In the co-verification environment, emulation speed is reasonable and it is able to fulfill all metrics including interface coverage.

We modeled peripheral interface module between API functions of FPGA card and virtual ARM cores. It makes big latency and decreases whole verification system performance. In case of using real ARM board, physical communication interfaces between hardware engines and host PC including RS232, DPP affects to whole verification system performance. In particular, if large data is applied as input of system, narrow bandwidth buses significantly affect the whole execution time and verification system performance.

We evaluate the verification time according to the kind of verification environments such as HDL simulator, co-simulation environment, and co-verification environment. Figure 8 shows the comparison results between these different verification environments according to the complexity of target design. Co-simulation environment shows the superior verification speed. The verification time of co-simulation environment is 80% and 170% faster than that of HDL simulator and co-verification environment, respectively. The verification time for co-verification environment is time consuming comparing to the other counterparts until the hardware complexity becomes 4M gates. Co-verification environment shows the higher verification speed than HDL simulator at this time. In addition, it is to overtake the verification speed for co-simulation environment after the complexity of target design is around 6.2M gates. Consequently, the verification time for co-verification environment for the target design consisting over 10M gates is 200 and 100,000 times higher than co-simulation environment and HDL simulator, respectively. This result notes that co-verification environment employing H/W emulator including HW accelerator and embedded processor for S/W design is preferred to the other verification environment when the complexity of target hardware-software co-design SoC is over 6M gates.

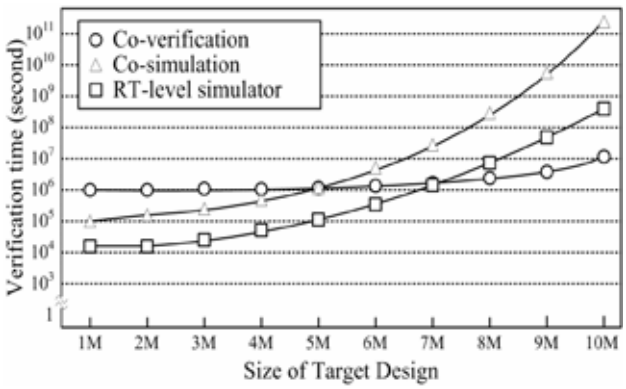

Fig. 8. Comparison result of verification speed according to the complexity of target design.

5. CONCLUSIONS

We conclude that combination of platform based design methodology and some co-verification methods is effective solution for today's SoC design. But it is important to accurately plan verification for the design in early phases of overall design process. Verification plan must consider all of aspects of design in each abstraction levels involved by design process. For example, when design needs various kinds of hardware components and engines, overall verification system performance is significantly affected by intercommunication ability between sub components and engines used in design. Thus, when designer makes an environment that is used to create virtual environment of SoC platform for various complex design, designer must carefully consider aspect of intercommunication between HW and SW design environment as possible as high flexibility. A poor intercommunication capability brings difficulties of design and verification at higher abstraction levels. Another important aspect is choosing what kind of verification model is required to meet the objectives of your design verification. This paper suggests some kinds of verification method we use for SoC design. The IEEE 802.11a compliant WLAN SoC is verified in the flexible development SoC platform. As a result, the target SoC design is successfully implemented and functionality is demonstrated. The proposed verification methodology gives an efficient link technique between the different verification levels.

REFERENCES

[1] J. Bergeron, Ed., *Writing Testbenches: Functional Verification of HDL Models,* Kluwer Academic Publishers, London, 2000.

[2] Y. Nakamura, K. I. Hosokawa, I. Kuroda, K. Yoshikawa, and T. Yoshimura, "A fast hardware/software co-verification method for system-on-chip by using C/C++ simulator and FPGA emulator with shared register communication," *Proc. of DAC*, pp. 299-304, 2004.

[3] S. Yoo, G. Nicolescu, L. Gauthier, and A. Jerraya, "Automatic generation of fast timed simulation models

for operating systems in SoC design," *Proc. of DATE*, pp. 620-627, 2002.
[4] J. Bieger, S. A. Huss, M. Jung, S. Klaus, and T. Steininger, "Rapid prototyping for configurable system-on-chip platforms – a simulation based approach," *Proc. of the 17th Int'l Conf. on VLSI Design*, pp. 577-584, 2004.
[5] T. Li, S. Li, J. Yu, and Y. Guo, "A novel collaborative verification environment for SoC co-verification," *Proc. of the 11th Int'l Conf. on Computer Supported Cooperative Work in Design*, pp. 145-150, 2007.
[6] Dynalith Systems, iPROVE: A Block Design and Verification Platform, 2003.
[7] K. Ahn, S. Kim, J. Kim, and C. Min, Implementation of a flexible development platform for simultaneous support of software and hardware development flow, *Proc. of ASICON*, Vol. 2, pp. 881-885, 2005.
[8] Milan Saini and Ross Nelson, "Today's platform FPGA systems require a proven co-verification methodology," http://www.mentor.com/products/fv/techpubs/
[9] H. Lee, J. Lee, S. M. Kim, and K. Cho, "Implementation of IEEE 802.11a Wireless LAN," *Proc. of 3rd Int'l Conf. on Convergence and Hybrid Information Technology*, pp. 291-296, 2008.
[10] Dynalith Systems, *Probase : A Block Design and Verification Platform*, http://www.dynalith.com, 2003.

Je-Hoon Lee
He received the B.S, M.S in computer and communication engineering from Chungbuk Nat'l University, Korea in 1999 and 2001 respectively and also received Ph.D in computer and communication engineering from Chungbuk Nat'l University in 2005. From 2005 to 2006, he was a visiting scholar at Univ. of Southern California, USA and from 2007 to 2008, he was a visiting scholar at Murdoch University, Australia. From 2006 to 2009, he was an assistant Professor in Chungbuk Nat'l University. He is currently an assistant Professor in Div. of Electronics and Information Communication Engineering of Kangwon Nat'l University. His research interests include embedded system applications high-speed and low-power circuit and system design.

Sang Choon Kim
He received the B.S. in computer science from Hanvat Nat'l University, Korea in 1986 and received M.S. in computer science from Cheongju University in 1989, and also received Ph.D in computer science from Chungbuk Nat'l University in 1999. From 1983 to 2001, he worked in ETRI. Since 2001, he has been on the faculty of the Kangwon National University at department of Information and Communication Engineering. His research interests include application security, security evaluation, IPTV security and network security.