UDC 621.3.049.771.14.001.63

Verification Methodology for a Complex System-on-a-Chip

●Akihiro Higashi ●Kazuhide Tamaki ●Takayuki Sasaki

(Manuscript received December 1, 1999)

Semiconductor technology has progressed to the point where it is now possible to implement system-level functions on a single LSI chip. However, traditional LSI verification becomes less and less powerful as the scale and complexity increase. In fact, more than half of the time required to develop a System-on-a-Chip (SOC) is used for function verification. A new verification methodology for SOCs should therefore be established.

We developed a system-level simulation technology to verify the specification and architecture of an SOC and a logic emulation technology to verify the logic function of an entire SOC. By combining these technologies, we established a powerful verification methodology for an SOC. We applied the verification methodology to develop a high-definition MPEG2 decoder LSI for a digital TV broadcasting system. The LSI was successfully developed on schedule and worked in the first silicon implementation completely according to the specifications.

1. Introduction

In the era of the System-on-a-Chip (SOC), we are now able to integrate the functions needed for consumer products such as digital home electronic appliances and advanced mobile devices on a single LSI chip. An SOC LSI includes complex functions with millions of logic gates. However, it is difficult to verify an SOC effectively using the traditional LSI verification methodology.

One of the most common applications of SOCs is in a video decoder system. It took almost two years to develop a Moving Picture Experts Group Phase 2 (MPEG2) decoder LSI. The main time-consuming tasks were the logic design/verification (eight months) and several revisions (10 months). In fact, it took a software logic simulator 10 hours to simulate a single video frame. Since a video stream contains 30 frames per second, it is difficult to develop a fully debugged video processing LSI using traditional LSI design/verification. The LSI was revised several times because of this difficulty.

We therefore examined the bugs that occur during LSI development and identified the following causes:

- Specification problems (insufficient definition, lack of necessary conditions, and misunderstandings between people)
- Implementation problems (insuficient performance, improper block partitioning, block interface mismatching, and excessive power consumption)
- Verification problems (slow software simulation and problems with the hardware-software interface and system function verification)

A new design methodology by which an SOC can be efficiently designed and verified should be established to overcome these problems. The design methodology for an LSI shifted from the transistor level in the 1970s to the gate level in the 1980s. Then, in the 1990s, it shifted to the

Register-Transfer Level (RTL), where logic circuits are described using a Hardware Description Language (HDL).

We have now established a new design methodology for SOCs. At the beginning of SOC design, we introduce a system-level simulation technique. A system-level simulation is performed using behavioral models written in C/C++ and is very fast because the behavioral models are highly abstracted to model entire system functions. The system-level simulation provides us with a powerful methodology to verify the specifications of a system and to check the architecture to realize a System-on-a-Chip.

We also introduce a hardware emulation system to verify RTL designs. Because a hardware emulator is 1000 to 10 000 times faster than a software simulator, we can simulate a frame of a video stream in only one minute using an emulator, whereas it would take 10 hours using a software simulator.

However, one of the drawbacks of using an emulation system is that it is hard to debug the circuit being emulated. That is, it is difficult to locate bugs in a circuit by checking the outputs of emulation results. To overcome this difficulty, we use the results of the system-level simulation mentioned above. Generally, an LSI is designed block-by-block. Therefore, if we compare the results of the system-level simulation with those of the emulation block-by-block, we can easily locate bugs.

System-level simulation for specification and architecture checking, hardware emulation for RTL design verification, and a combination of these two for a debugging environment provides us with a powerful design/verification methodology for developing an SOC. This paper describes the system-level simulation technology and the emulation technology we used to develop an SOC for a digital TV broadcasting system.

2. SOC design methodology

Since the specifications and architectures of an SOC are complex, there is a big gap between the system-level design and the logic design. Logical errors in an RTL can be verified by a logic simulator, but specification errors that occur at system design are fatal.

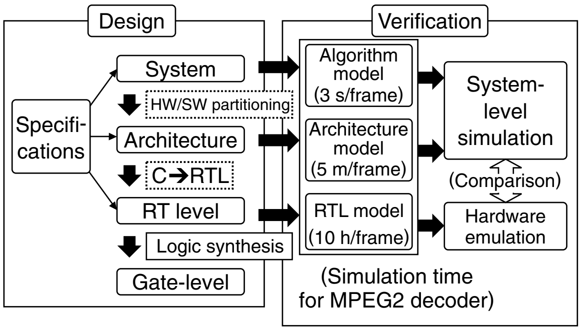


Figure 1
SOC design flow.

Figure 1 shows our design flow for an SOC. First, the designer defines the specifications of an SOC according to the industrial standards, required performance, and permissible costs. Usually, this is a paper plan. We develop the algorithm models for an SOC in C/C++ and verify the specifications and estimate the performance with system-level simulation. Many test scenarios are examined at high-speed by the system-level simulation.

Then, the architecture of the SOC is studied so that the specifications can be satisfied. The tradeoffs between the hardware processing and the software implementation for the SOC are also considered in the architecture design.

The architecture is studied using system-level simulation again. This time, however, the blocks of the SOC are modeled in cycle-accurate models. The cycle-accurate models include cycle-accurate timing and the exact data exchange among blocks. The simulation results are also used for reference when the RTL is being verified.

In the RTL design, the logic functions of the blocks are described in an RTL language and verified by RTL simulation. Then, they are integrated into a logic circuit for the SOC and simulated to verify the interfaces between blocks.

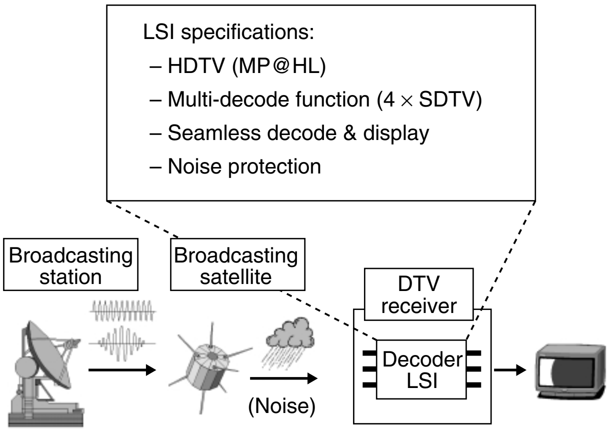


Figure 2
BS digital broadcasting system.

It takes a huge amount of time, however, to simulate an entire SOC using a software simulator. We therefore simulate the entire circuit using a hardware emulation system which runs 1000 times faster than a software simulator. However, it is not so easy to debug a circuit on an emulation system. We therefore compare the results of emulation with the results of the system-level simulation block-by-block, which makes it easy to locate errors in a circuit.

Our design/verification methodology for an SOC consists, therefore, of a system-level simulation for the specification and architecture check, RTL verification using a hardware emulator, and a comparison of the two sets of results.

3. SOC verification of digital TV decoder LSI

BS digital broadcasting will start this year (2000) in Japan to provide high-definition TV broadcasting. BS digital broadcasting will be provided to each home through the BS-4 broadcasting satellite. The video signals will be compressed according to the MPEG2 standard format, digitally modulated, and then transmitted. We applied our design methodology to develop a high-definition MPEG2 decoder LSI for the BS digital broadcasting receiver.[1)]

3.1 System-level simulation technology

Figure 2 shows an outline of the BS digital broadcasting system and the specifications of the MPEG2 decoder. The system-level simulation models the entire system at a high abstraction level and verifies the system performance at high speed.[2)]

We developed models for the specification stage and architecture design stage of the LSI's development. We also verified the specifications and estimated the performance by system-level simulation.

3.1.1 Specification simulation

We developed the specification models of the MPEG2 decoder LSI and the BS digital broadcasting system at the algorithm level in C language. The algorithm models for the MPEG2 decoder included a Transport Stream Decoder, MPEG2 (Main-Profile at High Level) Video Decoder, and Display. We checked the models by testing about 60 kinds of video streams.

Algorithm models of the BS digital broadcasting system were also developed to conform to the BS digital broadcasting specifications recommended by the Association of Radio Industries Business (ARIB). These specifications are transmission specifications, for example, for program data multiplexing, Reed-Solomon encoding, interleaving, Trellis Coded 8 Phase Shift Keying (TC8PSK), Quaternary PSK (QPSK), and Binary PSK (BPSK), for a broadcasting station and for receivers on the ground.[3)]

Figure 3 shows the block diagram of the BS digital broadcasting system that was used for the specification simulation. The specification simulation enables us to simulate many complicated combinations of broadcasting standards and reception conditions (e.g., rain and lightning).

3.1.2 Architecture simulation

Next, we developed detailed models at the behavioral level to verify the SOC's architecture, for example, its memory controls and scheduling. There are several important requirements when developing models for system-level simulation: mixed-level models must be suitable for mixed-level simulation, the block interface must be

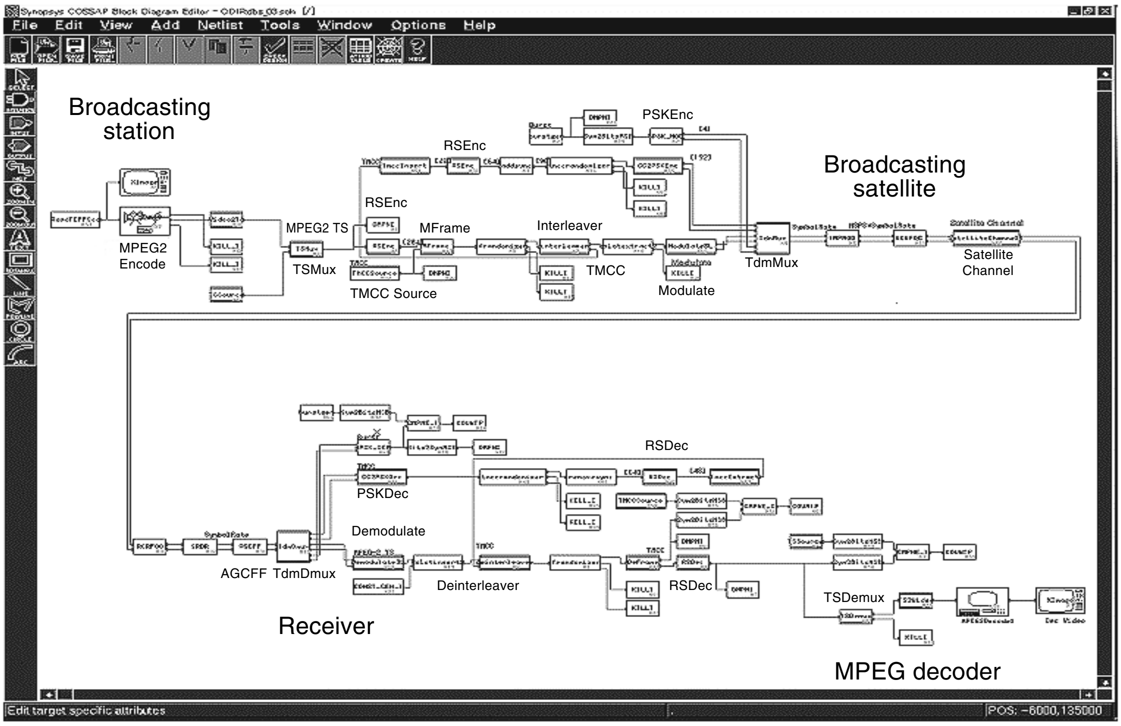


Figure 3
Block diagram for specification simulation.

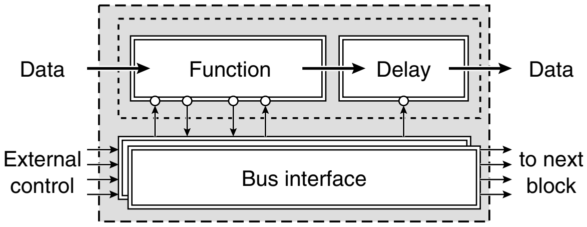


Figure 4
Simulation model.

flexible, and the simulation must be fast.

The architecture models include the facilities for structure and timing control. Each model is composed of a Function unit, Delay unit, and Bus interface unit, as shown in **Figure 4**.

The Function unit describes the block's function, the Delay unit describes the cycle delay operation for timing control, and the Bus interface unit is a variable precision model for hierarchical use.

In the architecture-level system simulation, we studied the LSI's circuit partitioning, memory buffer size, bus architecture, pipeline stages for data processing, and other details. It took five minutes to simulate one frame of video in the architecture-level simulation, whereas it took three seconds in the specification-level simulation: the more complicated the model, the more time the simulation took.

The results of the architecture-level simulation are also referenced in the RTL verification. The blocks in the architecture-level simulation correspond block-by-block to those of the RT-level simulation. Errors in a block could easily be detected by comparing the outputs of a block at the RT level with the outputs at the architecture level.

3.2 Emulation technology

Emulation systems are recognized as useful tools for detecting logical errors at high speed. However, it is generally difficult to obtain a high performance from an emulation system. First, it takes a long time to get things ready for emulation; emulation can be executed only after all of the RTL blocks are available. Since we did not have much emulation time before the scheduled delivery of the LSI's mask data to the manufacturing process, we had to start emulation as soon as possible. A second difficulty with emulation is that the emulation speed depends on the emulation execution mode. Therefore, to achieve the maximum speed in an emulation system, a dedicated environment is required. A third problem is that although emulators are very fast, their debugging environment is not as sophisticated as that of a software simulator, so it is difficult to use them to detect bugs efficiently.

We solved these problems with verification using an emulation system and then used the emulation system to perform a running test of the MPEG2 decoder LSI. The design verification flow in the emulation is shown in **Figure 5**. The LSI is verified by looking at the pictures of a video stream which were decoded by emulation. The verification environment is shown in **Figure 6**. The target circuit of the LSI and a test-bench is loaded onto the emulation system, and an emulation is executed in a mode called the synthesizable testbench (STB) mode.

Emulation in the STB mode has several advantages: the execution speed is very high, there is no need to develop verification devices to use the emulation system, and there is no need to consider the physical obstacles to debugging the LSI. However, there are also disadvantages. For example, the testbench for a software simulation cannot be reused on an emulation system. This means that we have to develop another testbench for emulation. In the STB mode, the RTL codes for a testbench must be able to synthesize down to a logic gate circuit.

To develop a sysnthesizable testbench, we prepare the following reusable design elements: SDRAMs, SRAMs, and a simple CPU. We developed RTL model generators for the SDRAMs and SRAMs. The generators can generate various SDRAM and SRAM models with optional memory capacities and word-lengths. The generators enable us to build the SDRAM and SRAM models on demand within a single day.

We also attached a debugging scheme to an SDRAM model so that it could be efficiently debugged on the emulation system. Because SDRAMs have complicated memory controls and the procedure for accessing an SDRAM is complicated, circuit designers tend to make errors regarding the command access, and these errors are difficult to detect. The error detection scheme in the SDRAM model checks whether the sequence

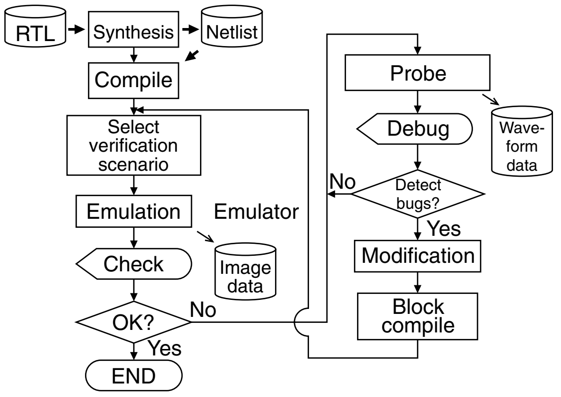
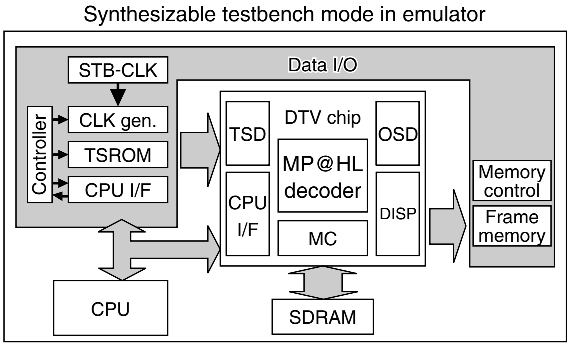


Figure 5
Verification flow in the emulation.



STB-CLK : Emulator master clock
TSROM : Transport stream ROM
OSD : On-screen display
CLK gen. : Clock generator
TSD : Transport stream decoder
MC : Memory controller

Figure 6
Verification environment.

of commands to an SDRAM satisfies the specifications. Then, any command access errors are automatically detected during emulation.

3.3 Results

3.3.1 Results of system level simulation

Figure 7 shows some pictures that were decoded by the system-level simulation. The pictures were transmitted over a noisy channel. Broadcast waves are affected by rain, lightning, and the noise between channels. Figure 7 (a) shows a decoded picture after noise processing, and Figure 7 (b) shows the same picture with no noise processing. In a digital TV system, noise data may cause TV receivers to hang up, so appropriate noise processing is essential in a digital TV receiver. We can check noise processing algorithms by system-level simulation. By viewing decoded pictures such as the one shown in Figure 7, we can directly check the effect of the noise processing algorithms.

Table 1 shows the execution times for the algorithm-level, architecture-level, and RTL-level simulations. The specification and architecture models for the MPEG2 decoder were written in C language. Simulation took three seconds per frame at the algorithm level, five minutes per frame at the architecure level, and 10 hours per frame at the RTL level. The table shows that system-level simulation greatly reduces the time needed for circuit verification.

In the past, we could not see the decoded pictures until the RTL design had been completed. Now, using system-level simulation we can check the specifications and architecture at an early stage of LSI design. As a result, we do not need to perform time-consuming RTL re-design and the risk of the need to perform LSI re-makes is reduced. Thus, we can reduce the time and cost of LSI development.

3.3.2 Emulation results

After the RTL design, we verified the entire circuit on two emulation systems. The running test was begun one month after all RTL blocks became available. We used two emulation systems: CoBALT by Quickturn Design Systems and Celaro by Mentor Graphics Corporation.[4),5)]

The circuit has multi-rate clocks: 125 MHz, 125/2 MHz, 125/4 MHz, 72 MHz, 72/2 MHz, 33 MHz, and Transport Stream Rate (variable). The emulation speed was from 105 to 144 kHz, though the asynchronous multi-clock system tends to slow down the emulation speed. If we include the time needed to download the emulated frames, the emulation speed was from 45 to 47 kHz.

During the LSI development, more than 30 video streams were emulated and more than 6000 frames were decoded. About 300 frames were emulated per night. The emulation detected more than 30 bugs, and some bugs were detected after emulation of 100 frames. Also, bugs in the order of SDRAM commands were detected. It would

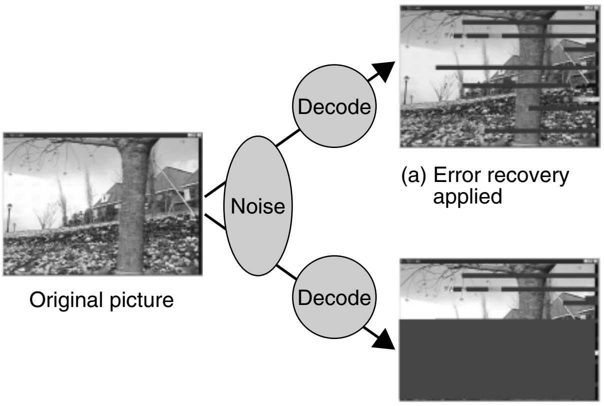


Figure 7
DTV system simulation.

Table 1
System simulation performance.

Simulation model	Performance
Algorithm model	3 s/frame
Architecture model	5 m/frame
RTL model (synthesizable)	10 h/frame

note) Target : High-definition digital TV decoder LSI
Simulation time : Ultra SPARC 250 MHz

have taken about one month to detect such crucial bugs as these using a software simulator, so the detection scheme for the order of SDRAM commands was very useful.

4. Conclusions

We have developed a system-level simulation technology to verify the specification and architecture of an SOC and a logic emulation technology to verify the logic function of an entire SOC. By combining these technologies, we have established a powerful verification methodology for SOCs.

We applied the verification methodology to develop a high-definition MPEG2 decoder LSI for a digital TV broadcasting system. In the past, we could not see the decoded pictures until the RTL design had been completed. Now, we can check the specifications and architecture at an early stage of LSI design by performing the system-level simulations. As a result, we can avoid time-consuming RTL re-design and reduce the risk of LSI re-makes.

During the LSI development, more than 30 video streams were emulated and more than 6000 frames were decoded. Frames were emulated at the rate of about 300 per night. More than 30 bugs were detected by emulation, and some bugs were detected after 100-frame emulation.

We then completed the design of our MPEG2 decoder LSI for a digital TV broadcasting system. The LSI was developed on schedule, and the first silicon implementation worked completely according to the specifications.

References

1) H. Takahashi, Y. Otobe, and K. Kohiyama: Single-chip MPEG2 MP@HL Decoder with Multi-decode and Seamless Display Features. *FUJITSU Sci. Tech. J.*, **36**, 1, pp.48-55 (June 2000).
2) A. Higashi: System LSI Verification Technology with System Level Simulation User's Meeting'98. Synopsys. Tokyo, 1 Dec. 1998.
3) H. Matumura: Digital Satellite Broadcasting System. (in Japanese), Proceedings of NHK STRL (Science & Technical Research Laboratories) OPEN HOUSE'98, 1998, pp.1-16.
4) Quickturn: CoBALT User's Guide, version 1.1 edition, 1997.
5) Mentor Graphics: Celaro User's Manual, software version 2.3_1 edition, 1999.

Akihiro Higashi received the B.S. degree in Electronics and Communications Engineering from Meiji University, Tokyo, Japan in 1983. He joined Fujitsu Ltd., Kawasaki, Japan in 1983, where he was engaged in research and development of home information terminals and their ICs. He was transferred to Fujitsu Laboratories Ltd., Kawasaki, Japan in 1993, where he has been engaged in research and development of ICs for image processing systems. Since 1996, he has been engaged in research of design methodologies for system-on-a-chip. He is a member of the Institute of Image Information and Television Engineers (ITE).

Kazuhide Tamaki received the B.S. degree in Production Engineering from Nihon University, Tokyo, Japan in 1987. He joined Fujitsu Ltd., Kawasaki, Japan in 1987, where he was engaged in research and development of home information terminals and their ICs. He was transferred to Fujitsu Laboratories Ltd., Kawasaki, Japan in 1993, where he has been engaged in research and development of ICs for image processing systems. Since 1996, he has been engaged in research of design methodologies for system-on-a-chip.

Takayuki Sasaki received the B.E. and M.E. degrees in Information Science and Engineering from Tokyo Institute of Technology in 1995 and 1997, respectively. He joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 1997, where he has been engaged in research of design methodologies for system-on-a-chip.