Hardware–Software Codesign for Graphic LSIs

● Hirohisa Kotegawa ● Naonobu Hasumi

The field of image processing LSIs that are mounted in devices such as digital cameras is one in which the improvement in performance is very remarkable. It is important to have an architecture design that is suitable for the application so as to lower the cost of developing chips for LSIs, and reduce their power consumption and the chip area. Semiconductor and EDA vendors have proposed hardware-software (HW/SW) codesign via electronic system level (ESL) as a way to develop chip architecture and designs. However, it has not been used much in actual development sites because of various issues such as the cost of developing models. Under these circumstances, at Fujitsu Semiconductor we have steadily applied architecture design technology to LSI development based on HW-SW coverification in the ESL methodology in Cedar service, which is a design service for customers to develop ASSPs and ASICs. In this way, we have found that the most important point is to optimize the quality of service (QoS) of on-chip buses and access to external memory such as DDR memory. And, as a result of focusing on this point of optimization and investigating ways to tackle the issues with ESL, we have created a new HW/SW codesign solution that uses ESL. It has come to be used in design sites more than ever before. This paper introduces the technical aspects of this new approach and its effect, and also describes future developments.

1. Introduction

Catalogs of digital cameras and other products that integrate image-processing LSIs[1)] often show descriptions such as the effective pixel count of CCD and CMOS sensors being 10 megapixels. These pixel counts are increasing year by year and, with some of the top-class models of single-lens reflex cameras, the counts may exceed 30 gigapixels. In addition, the degree of complexity is increasing not only in image processing itself, including various types of image correction, image effect and image scaling, but also in functions such as graphical user interfaces using Android OS and various external interfaces. For this reason, circuit scales are expanding due to the enhanced functionality and performance of CPU cores and intellectual property (IP) cores that constitute imaging LSIs and increase in RAM size, which has also caused increased LSI power consumption.

While dealing with these increases in circuit scales and power consumption, mobile products such as digital cameras must satisfy requirements including minimization of chip areas and reduction of power consumption in view of battery life and packaging cost, and these are in a trade-off relationship with the scale of circuits. Up to now, these requirements have been handled by integration and voltage reduction technologies. However, catching up with Moore's law is becoming difficult with the recent cutting-edge process technologies, and improving power consumption is also growing increasingly difficult due to limitations of voltage reduction and increased leakage current and wiring capacity. Accordingly, evaluating architecture to decide whether performance, power consumption and chip area have been optimized in line with the system use case in the architecture design phase, which is an upstream process, is gaining importance.

In reality, however, architecture evaluation in an upstream process has not spread much in the actual design sites. The reasons for this include, in addition to the difficulty in establishing an environment that

facilitates architecture evaluation in an upstream process, the development flow does not allow architecture evaluation. Generally, an image-processing LSI is developed as a system-on-a-chip (SoC), which integrates various IP cores such as one or more CPU cores, graphics processing units (GPUs) and digital signal processors (DSPs) into a single chip. For that reason, operation as an image processing system cannot be realized without installing the software. This means that both software and an SoC are required for evaluating the performance and power consumption as a system. In the actual development flow, however, software is generally developed on a production board after an engineering sample (ES) of an SoC has been made and software does not exist in the architecture design phase. That is why architecture evaluation of an SoC has been impossible in an upstream process.

Fujitsu Semiconductor has built a hardware-software (HW/SW) codesign flow that makes use of electronic system level (ESL) techniques to address this issue of inability to evaluate architecture.

This paper describes this HW/SW codesign.

2. Outline of and issues with ESL

Before going into the main theme, this section describes ESL and the abstraction level of models and presents issues that have hindered the application of ESL up to now.

2.1 What is ESL?

ESL refers to an environment in which an SoC is modeled by using a hardware description language such as SystemC based on C/C++ to conduct virtual simulation on a computer that is used for advanced development of software and architecture evaluation.[2)] In the electronic design automation (EDA) industry, ESL may also refer to using a hardware description written in a language such as SystemC to generate a register-transfer level (RTL) description by high-level synthesis but, in this paper, it means the simulation environment mentioned above.

2.2 Abstraction level of models

In this way, ESL is used for two purposes, namely advanced development of software and architecture evaluation, and the modeling method may differ depending on the purpose.

Software development requires a simulation performance closer to that of a production device and modeling with a high abstraction level is necessary. For architecture evaluation, data read and write must be carried out in a manner similar to the actual hardware communication protocol, which requires modeling with a low abstraction level. Modeling methods suited for different applications have been standardized as transaction-level modeling (TLM) by the Open SystemC Initiative (OSCI), a systemC standardization organization. In TLM, a model with a high abstraction level suitable for software development is called a loosely timed (LT) model and that with a low abstraction level suitable for architecture evaluation is called an approximately timed (AT) model.

2.3 Issues with ESL

In this way, environments with two degrees of abstraction – LT and AT – must be prepared according to the application as the ESL environment, and this accordingly requires person-hours and money to establish. In addition, development of a model with a low abstraction level used for architecture evaluation needs person-hours equivalent to that for RTL development and it may seem like duplicate development unless an RTL description can be generated by high-level synthesis or other means. This problem becomes conspicuous in SoC development because RTL descriptions often use IPs that already exists. Furthermore, with IPs introduced from other companies or legacy IPs, the specifications required for modeling are unclear and modeling is practically impossible in many cases. That has raised the barrier to introducing models in the actual design sites and hindered acceptance.

3. Points of evaluation of architecture design

To overcome the issues described in the previous section, Fujitsu Semiconductor has utilized its past experience to sort out important points of evaluation in architecture design and taken measures. This section describes those points of evaluation.

In LSI development up to now, there have often been cases in which IPs as they are meet the performance requirements but, after they are built into SoCs, they fail to satisfy the requirements.

Focusing on such difficulties, we conducted

analysis while reproducing the conditions that caused problems by such means as emulation. As a result, we have found out that on-chip buses relating to IPs and external memory controllers for DDR and other memory are unable to optimally handle data and other IPs in contention in terms of data flow cause hindrance, which obstruct performance enhancement. That is, it has been revealed that the primary cause of hindering performance enhancement is the failure to optimize on-chip buses and external memory controllers for DDR and other memory and that their optimization is key to improving performance.

Based on this result, we have assumed AXI (AMBA3) interconnect bus, which is currently often used for the on-chip buses of Fujitsu Semiconductor's image-processing LSIs, and, of the external memory controllers, DDR memory controller, which requires read/write latency optimization by various types of command control and identified specific points of evaluation, as shown in **Table 1**.

4. HW/SW codesign flow

Table 1 also lists the points of evaluation and degrees of abstraction of the design environment and input required for the evaluation. In view of those points, we have established a new HW/SW codesign flow, which is shown in **Figure 1**. This section describes the respective environments that constitute this design flow.

First, to focus on the abstraction levels of design environments in Table 1, a level called cycle accurate (CA), which has not been common as an abstraction level of models in ESL up to now, is included. This abstraction level is equivalent to RTL and even lower than AT and more faithfully represents operation of the actual hardware. Based on Fujitsu Semiconductor's past experience in architecture evaluation, it has been found that the CA abstraction level is required for bus arbitration to handle AXI requests in contention and command control (ordering control for hiding precharge and refresh control) to effectively use the memory channel bandwidth of DDR memory controllers. Otherwise, operation significantly differs from the actual operation, which considerably distorts the results of architecture evaluation. However, not all of the abstraction levels of design environments for the respective points of evaluation in Table 1 have to be CA. As described earlier, a lower abstraction level means a lower simulation speed and it is impossible to execute software in a manner similar to that with an SoC in full configuration. Accordingly, we have built a

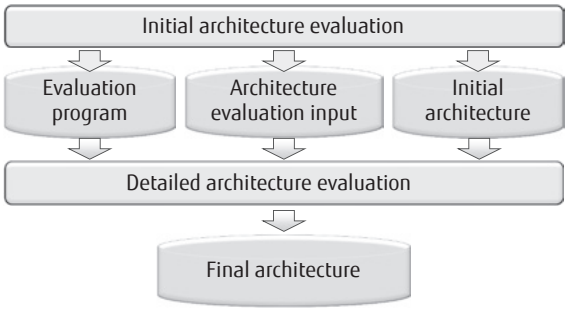

Figure 1
HW/SW codesign flow.

Table 1
Points of performance evaluation.

Point of evaluation		Requirement	Abstraction level
On-chip bus	Topology	System operation	AT
	Operating frequency		
	Number of FIFO stages	Data flow	CA
	Arbitration		
	Priority control		
DDR memory controller	Number of ports	System operation	AT
	Operating frequency		
	Number of FIFO stages	Data flow	CA
	Arbitration		
	Command control		

design flow combining two evaluation environments: initial architecture evaluation environment, in which the AT abstraction level is used to determine the base architecture and obtain information required for the subsequent evaluation; and detailed architecture evaluation environment, in which the architecture is finally determined with the CA abstraction level.

4.1 Initial architecture evaluation environment

In the initial architecture evaluation environment, the points of evaluation to be evaluated with the AT abstraction level in Table 1 are evaluated. As shown in **Figure 2**, this environment is characterized by the use of LT models for hardware that is less important for performance evaluation but necessary for operation as a system, such as the peripherals, and use of AT models only for hardware that is important for performance evaluation such as the CPU cores and AXI bus. In addition, those that cannot be modeled by Fujitsu Semiconductor such as GPUs and other IPs introduced from outside and customer logic are mapped to an FPGA and the SCE-MI technology is used to allow high-speed connection between the FPGA and ESL on the PC. In this way, we have minimized AT models, which degrades the simulation performance, and mapped to an FPGA and connected IPs that are difficult to model. This has made it possible to evaluate architecture as a system by using evaluation software close to an application in a reasonable manner in terms of simulation performance and development person-hours.

In this environment, the basis of the detailed architecture is determined including the AXI topology, memory size required for the DDR and specification of the DDR memory controller interface. In addition, information required for evaluation in the detailed architecture evaluation environment can be obtained including the data flow needed for identifying performance bottlenecks, which provide points of performance evaluation, and evaluating the bottlenecks and related master transaction behavior.

4.2 Detailed architecture evaluation environment

As a result of evaluation in the initial architecture evaluation environment, important master and slave IPs are identified based on the performance bottlenecks and related data flow. Then, RTL models of the IPs are converted with an EDA tool equipped with a technology called carbonization, which is capable of conversion into C models with the CA abstraction level maintained. With the C models resulting from the conversion, a simulation environment that is five to ten times faster than RTL can be built. However,

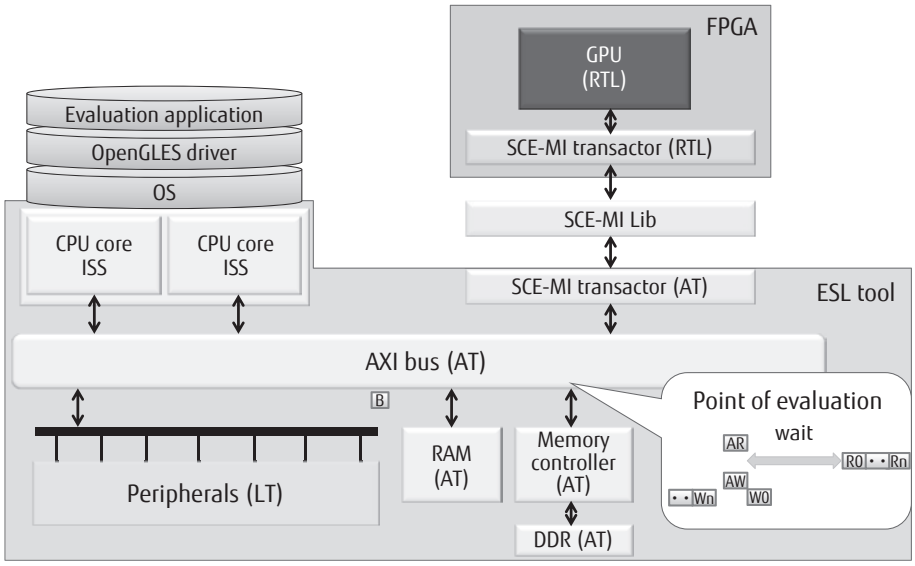

Figure 2
Environment for initial architecture evaluation.

when conversion into C models is difficult due to the circuit scale or microarchitecture or when a few tens of seconds are required as the real time of the actual application operation for running the IPs and realizing the data flow that causes a bottleneck, the environment cannot be practically used even if it is faster than RTL by five to ten times. This problem can be solved by using a general-purpose transaction generation model, which is shown in **Figure 3**. This model is capable of flexibly controlling transaction generation in the C language, which allows easy generation of a transaction that causes a bottleneck and representation of the state of a performance bottleneck as a system more easily than connecting the actual IPs.

Evaluation in this environment for detailed architecture evaluation has made it possible to make final adjustments including determination of the number of FIFO stages and arbitration of the AXI bus and the adjustment of the number of command queue stages and determination of the command control method of the DDR memory controller in a short time before the entire RTL is built up.

5. Effect of application

By using the HW/SW codesign flow that utilizes this new ESL technology, performance problems, which could not be detected until the emulation or production device evaluation phase, have now become detectable in the architecture design phase.

With AXI and other protocols, requests and data are independently controlled and throughput and latency check with a single transaction alone often does not reveal problems. As in image processing, performance problems may not be exposed before multiple image frames have been processed. To address this issue, long-time simulation of about 10 seconds in real time can be conducted as the initial architecture evaluation to identify the problem trends in multiple-frame processing and the problems can be simulated in the detailed architecture evaluation environment. In this way, essential causes of performance problems can now be detected in a short period of time in the phase of architecture specification establishment and can be avoided.

As a specific example, there was a case in which a legacy IP that had poor response but did not cause any performance problem as it was, was run against another IP that contended with it in the data flow and processed a large amount of data. Over time, the poor response gradually caused an adverse effect, and eventually led to performance degradation. This problem was detected and solved in the specification phase and the image-processing LSI developed in this way was put into full operation without any issues.

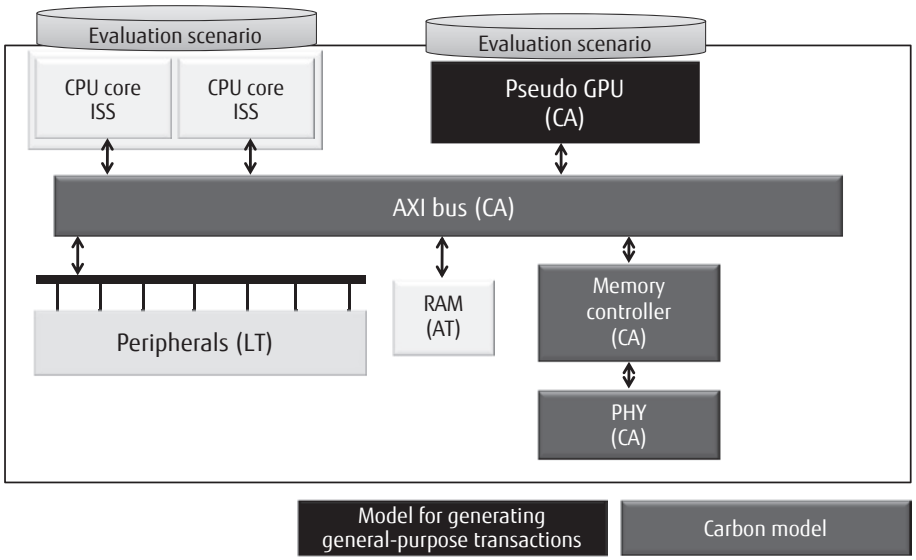

Figure 3
Environment for detailed architecture evaluation.

6. Future issues

As described above, the HW/SW codesign flow making use of the ESL technology has started to produce tangible results but it still has the following issues.

1) Modeling of outside of high-speed interface IPs

High-speed interface macros such as USB and PCI Express macros are connected with the outside of an SoC in various use cases. For that reason, it is almost impossible to model the outside in view of use cases. At present, the worst-case pattern of the use of the SoC is assumed and the general-purpose transaction generation model mentioned earlier is used for simulation. However, it may lead to excessive performance, circuit scale expansion or increased power consumption caused by having too many transactions compared to the actual use case or, conversely, performance degradation due to having too few transactions.

2) Feedback to low-power design

Optimization of architecture design by the HW/SW codesign flow presented in this paper is believed to help reduce power consumption in terms of preventing excessive performance. However, how much optimization was achieved in terms of the actual power value has not been quantitatively visualized and not prepared as design data to be fed back to RTL implementation or layout design.

In the future, we intend improve the HW/SW codesign flow to resolve the two issues described above.

7. Conclusion

The HW/SW codesign flow presented in this paper has come to be applied to various ASSPs and ASICs in addition to image-processing LSIs. We are committed to making continued efforts to utilize the valuable feedback from customers of ASICs and designers of Fujitsu Semiconductor so that we can establish an even better upstream design flow.

References

1) H. Kaneko: Image Processing LSIs Supporting a Whole Range of Products from Digital Cameras to Mobile Phones behind the Scenes. (in Japanese). *http://i.impressrd.jp/files/images/bn/pdf/im200602-108-solution.pdf*

2) M.Omura et al.: VLSI Design by C/C++. (in Japanese), Kyoritsu Shuppan, 2003.

Hirohisa Kotegawa
Fujitsu Semiconductor Ltd.
Mr. Kotegawa is currently engaged in development of upstream design verification environment and assistance with upstream design verification.

Naonobu Hasumi
Fujitsu Semiconductor Ltd.
Mr. Hasumi is currently engaged in development of upstream design verification environment and assistance with upstream design verification.