Integrating UML into SoC Design Process

Qiang Zhu, Ryosuke Oishi, Takashi Hasegawa, Tsuneo Nakata

Integrating UML into SoC Design Process

Qiang Zhu Ryosuke Oishi
Fujitsu Laboratories LTD., Japan
{shiyu, roishi}@labs.fujitsu.com

Takashi Hasegawa
Fujitsu Limited, Japan
thasegaw@jp.fujitsu.com

Tsuneo Nakata
Fujitsu Laboratories LTD., Japan
nakata@labs.fujitsu.com

ABSTRACT

In this paper, we proposed a method for integrating UML model into the current SoC design process. UML is introduced as a formal model of specification for SoC design. The consistency and completeness of the specification is validated based on the formal UML model. The implementation is validated by a systematic derivation of test scenarios from UML model. The method has been applied to the design of a new media-processing chip for mobile devices. The application of the method shows that it is not only effective for finding logical errors in the implementation, but also eliminates errors due to inconsistency and incompleteness of the specification.

1. Why UML?

System-on-Chip (SoC) validation has become increasingly difficult with increasing its complexity and scale. Industry studies reveal that more than 50% of the total schedule is being spent in verification process; about 65% chips have to respin after finishing them. In our experiences we find many ambiguities, flaws and oversights appear in the specification written in the natural language by designers. We need a new methodology to improve the quality of the specification as well as the implementation.

The Unified Modeling Language (UML) is a modeling language for capturing the specification, the implementation using the graphical notations in software community.

We also think that the UML is a good choice to model the specification for SoC, because:

- UML describes the specification graphically so that it can be easy to help development members understand and review the functional specification of the target SoC system.
- UML models the specification from different views using different diagrams. This feature can help us to model the functionalities, data structures, architectures, behaviors with different diagrams.
- UML provides well-defined notations to describe the specification more formally than that being written in natural language. This is not only help us remove the ambiguities from specification but also lead us to develop a tool to verify the correctness of the specification and generate the test scenarios for the implementation automatically.

However UML also have some issues when we try to apply it for a SoC design.

- UML only defines many types of diagrams but does not describe how to use them. We must clarify that which part of the specification should be modeled, which diagrams should be used, and how to model the specification with different diagrams.
- UML only provides the definition of diagrams but not mentions how to integrate them into a design process. We must establish a methodology to integrate the UML into the current design process for SoC.
- UML have many diagrams for mainly capture static and dynamic views for the software system. However, the previous UML specification did not have the proper notations to model the protocol at signal level for interface communication of SoC. We need to find a good manner to describe the interface communication protocol at signal level.

In this paper, we show a method how to integrate the UML into the conventional SoC design process. The application results show our approach is not only useful for specification validation, but also is effective for implementation verification.

2. Integrating UML into SoC verification process

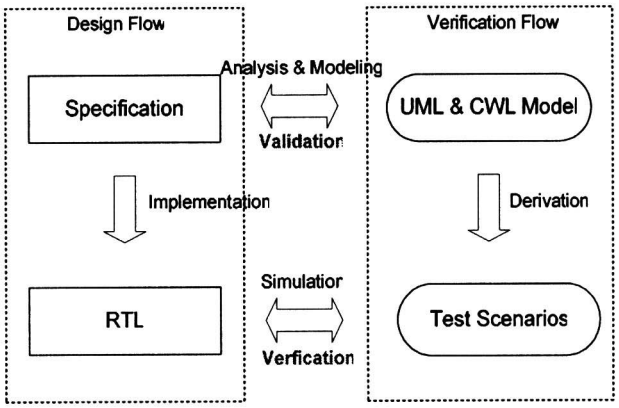

Figure 1: Integrating UML into SoC Design Process

The key strategies in our approach are shown in Figure 1.

Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'05)
1530-1591/05 $ 20.00 IEEE

- For reducing the impact to the current design flow, we integrate UML into the verification process without changing the current design style.
- We only utilize use case diagrams, sequence diagrams and class diagrams of UML to model functions, data types and behaviors in the specification.
- We introduce CWL (Component Wrapper Language) [2] to model the interface communication protocol at signal level.
- We use UML to analyze and model the specification written in natural language and eliminate the errors from the specification by validating the completeness and consistency of the UML model.
- We derive the test scenarios from UML model to validate the implementation (RTL source code) using the black box test.

The UML model can help us validate the correctness of the specification by checking the completeness and consistency of UML model includes [3]:

- **The completeness of use case** indicates that use cases must cover all functions in specification and each function in specification must be used by a use case once at least.
- **The completeness of event flows** indicates that the listed paths of event flow include all possible scenarios that can occur in the use case.
- **The consistency between a use case and its event flows** indicates all normal paths (basic path, alternative path) of event flows in a use case can reach the given post-condition under the pre-condition defined in a use case description.
- **The consistency among UML diagrams** indicates there are no violations among UML diagrams.

The UML model can also help us derive the test scenarios for the implementation verification.

- **The test scenarios at use case level** must cover all use cases from the use case diagram.
- **The test scenarios at event flow level** can be extracted from all paths of event flow include basic, alternative and exceptional paths from sequence diagrams.
- **The test scenarios at parameter level** can be extracted from class diagrams and their constraints.

3. Application

We used our approach for a Mobile Media Processors (MMPs), We organized a verification team separated from the design team to validate the specification written in natural language by designers and validated implementation with black box test. The UML and CWL were used as modeling languages to analyze and formalize the component specification and interface protocol specification respectively. The specification validation was performed by validating the completeness and consistency of the UML model. The specification was brushed up by eliminating those errors due to incompleteness, inconsistency in specification at analysis and modeling phase using UML. Then we derived test scenarios from the UML model and generated interface protocol monitors to verify the functionality and interface protocols from CWL.

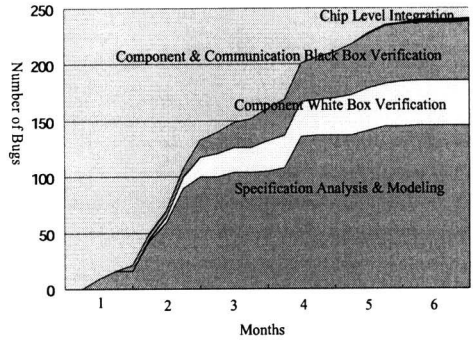

Figure 2: The results for errors and their discovery time

Figure 2 shows the results for errors and their discovery time. We found 132 errors that included oversights, mistakes, and ambiguities due to incompleteness and inconsistency in the specification at analysis and modeling phases. We also discovered 14 specification errors in the implementation test phase. After that, we found 51 errors from the implementation with black box test using the test scenarios and test benches derived from the UML model. Meanwhile, designers found 40 errors with white box test. At chip-level test phase, we only found 3 errors.

4. Conclusions

In this paper, we proposed a method how to integrate the UML into the current design process. We applied our proposal to a real design in Fujitsu. We found UML is not only useful for specification validation but also is effective for implementation verification. In future work, we will develop tools to validate the UML model and generate the test scenarios from UML model automatically.

5. Reference

[1] OMG home page, http://www.omg.org/

[2] Component Wrapper Language, http://www.labs.fujitsu.com/en/techinfo/cwl/index.htm

[3] Q. Zhu, R. Oishi, T. Hasegawa, T. Nakata, "System-On-Chip Validation using UML and CWL", Proc. of the International Conference on Hardware/Software Co-design and System Synthesis (CODES+ISSS 2004), pp. 92-97, Stockholm, Sweden, Sept., 2004.