Development of Automated Software Diagnostic Tool To Validate SOC-Based IPs

G.LAHARI (M Tech) Embedded Systems, AITS, RAJAMPET, INDIA

P.SIVA KALYANI, Professor, ECE, AITS, Rajampet, INDIA

lahari.gundam@gmaol.com

***Abstract**: In this project we will develop and implement a software diagnostic tool which can be used in post silicon validation to validate SoC IPs automatically. Post-silicon validation has become essential in catching hard-to detect, rarely-occurring bugs that have slipped through pre-silicon verification. Post-silicon validation flows, however, are challenged by limited signal observability, which impacts their ability of diagnosing and detecting bugs. Indeed, bug manifestations during the execution of constrained-random tests may be masked and be unobservable from the test's outputs. The ability to evaluate the bug-masking rate of a test provides great value in generating and/or selecting effective tests for high coverage regressions. To this end, we propose an efficient, automated software diagnostic tool development that can be used in post-silicon validation to validate various SoC based IPs like, UART, DMA, IIC, GPIO, Timers and many more. We will be developing test cases in C/C++ environment on Linux platform. The test cases can be executed randomly, rigorously, regressively can also be used to perform stressing on respective IP features. This might help in finding the bugs at SoC IP level, hitting the corner cases etc. And the same tool can also be used across the platforms, so that development time comes down, hence reduction in production cost.*

Keywords*: post silicon verification, pre silicon validation, UART, DMA, ARM based development board, JTAG programmer.*

I. Introduction

In semiconductor industry, the major cost involves in making the IC die, sometimes making of the die cost will be in millions of dollars. If any bug propagates in after making the die, it will impact huge loss for the industry. Hence, before actually making the die or before making the first piece of IC, it is required to verify the silicon design or a particular IP design thoroughly. The silicon industry follows this verification at various stages, broadly classified as verification and validation

Verification is a process of testing the design against a given specification before sending the silicon for production. This can be done using several methods like software simulations, synthesis, static formal analysis and FPGA/hardware emulation.

Validation is a process of verifying the real silicon or a particular IP, for all its specific functional and electrical correctness in a lab set up. This includes having the real chip assembled on a test board or a reference board, running real software/applications and making sure that all features work well.

The main advantages of using SoC are miniaturization size, cheapening product cost, reduced Time to Market. SoC is used in various well-known products such as cell phone, digital multimedia players, game console and other consumer electronic devices. SoC generally contains various cores that could be designed by different IP vendors.

II.EXISTING SYSTEM

In the currently available tools consumes manual efforts, and not automated. As these tools, are dependent on human efforts, finding the number of bugs depends on the skill set and patience of engineer that he is running manual tests cases. To avoid these kind of manual efforts, we are proposing Automated Software Diagnostic Tool development for SoC based IPs post silicon validation.

III.PROPOSED WORK

Advances in semiconductor technology enable us to integrate billions of transistors in a single chip. This ever-growing complexity poses a challenging problem for SoC based microprocessor or microcontroller design verification: indeed, reaching coverage closure within reasonable time is becoming extremely difficult. High-end SoC based microprocessors often include complex features (e.g., transactional memories, security enhancements and multiprocessor memory consistency) that are hard to verify in the early stages of verification. Therefore, post-silicon validation, that is, the validation effort carried out on the first silicon prototypes, aims at catching all the remaining bugs that were not detected in the pre-silicon stage. One of the most difficult aspects in the deployment of post-silicon validation, however, is the extremely limited observability into the design's internals.

Therefore, post-silicon validation, that is, the validation effort carried out on the first silicon prototypes, aims at catching all the remaining bugs that were not detected in the pre-silicon stage. One of the most difficult aspects in the deployment of post-silicon validation, however, is the extremely limited observability into the design's internals.

This aspect makes bug detection and diagnosis challenges of their own. Even more importantly, bugs may manifest during a test's execution, but become masked and go unnoticed by the time the test completes. The outcome in these situations is that the post-silicon test simply wastes precious prototype's execution cycles. The types of tests deployed in post-silicon validation vary widely, from application snippets, to compatibility tests, to constrained random tests. These latter ones are particularly valuable in trying to exercise corner-case scenarios, since a vast number of variants can be generated with little designer's effort. One additional benefit that they bring to the validation effort is that they can often be generated directly in the microprocessor under verification (i.e., on-platform), enabling an efficient use of the usually scarce number of platforms and prototypes available.

As part my curriculum project its very difficult carry out entire SoC validation, but my goal is to develop automated scripts using C/C++ on Linux platform for few IPs, as listed below.

1. UART
2. DMA
3. IIC
4. GPIOs
5. SPI

For each IP in above mentioned list, planning to develop minimum of 20 automated test cases. ARM Cortex based processor has been chosen to demonstrate the validation diagnostic software tool.

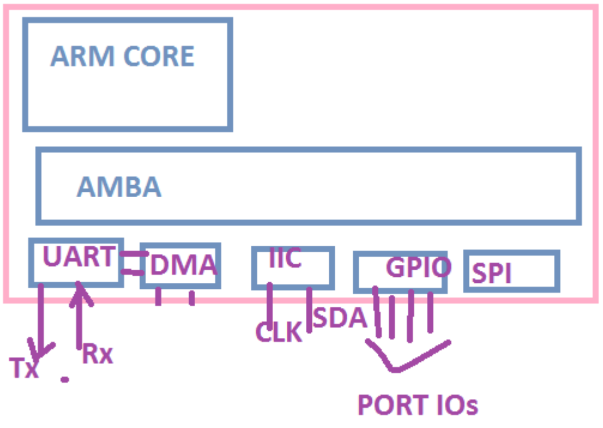


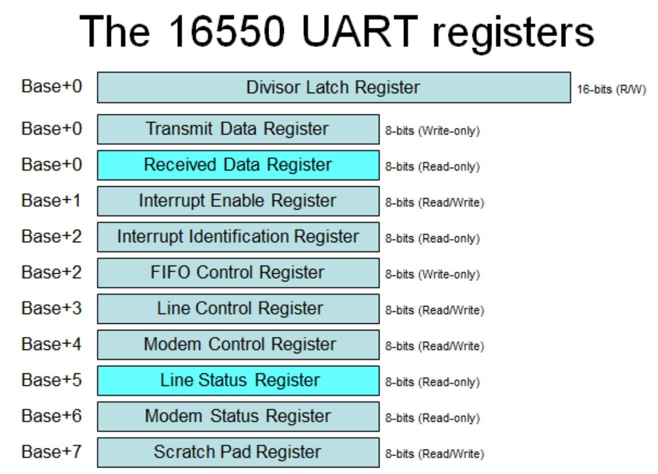
Figure: Typical view IPs inside SOC

A. UART DMA Validation

For each IP more than 20 features has been validated. The feature of UART DMA IP is listed below.

- DMA Interrupt Support
- UART Interrupt Support
- UART baudrate support based on internal clock
- DMA register access
- DMA fetching instructions
- DMA read transaction from system memory to UART FIFO
- DMA write transaction from UART FIFO to system memory.
- UART Controller register access
- Duplex data transaction on uart interface with auto flow enabled
- UART loopback feature
- UART Reference Clock Select register and DLH, DLL divisor programming
- UART MISC RX Control
- UART MISC TX Control

The list of registers in UART is as follows.



The 16550 UART registers

Base+0	Divisor Latch Register	16-bits (R/W)
Base+0	Transmit Data Register	8-bits (Write-only)
Base+0	Received Data Register	8-bits (Read-only)
Base+1	Interrupt Enable Register	8-bits (Read/Write)
Base+2	Interrupt Identification Register	8-bits (Read-only)
Base+2	FIFO Control Register	8-bits (Write-only)
Base+3	Line Control Register	8-bits (Read/Write)
Base+4	Modem Control Register	8-bits (Read/Write)
Base+5	Line Status Register	8-bits (Read-only)
Base+6	Modem Status Register	8-bits (Read-only)
Base+7	Scratch Pad Register	8-bits (Read/Write)

The algorithm for validation the feature is listed below.

Test Case1: Receive Buffer Register Access Test: UART0

Purpose: To verify the RBR access test on UART0.

Test Procedure:

- Reset UART and Wait till UART comes out of reset
- Configure UART baud using DLL and DLH registers
- Enable UART[1,0]x70[FAR]==1.
- FAR: FIFO Access Register. Read-write. Reset: 0. 1=Enable a FIFO access mode for testing when FIFOs are implemented and enabled; When FIFOs are not implemented or not enabled it allows the RBR to be written by the master and the THR to be read by the master. 0=FIFO access mode disabled.
- UART[1,0]x78 Receive FIFO Write (RFW).Write known data (lets say: 0xaa) into: RFWD (0:7): Receive FIFO Write Data in RFW register. Write-only. Reset: 0. These bits are only valid when UART[1,0]x70[FAR]==1. When FIFOs are implemented and enabled, the data that is written to the RFWD is pushed into the receive FIFO. Each consecutive write pushes the new data to the next write location in the receive FIFO. When FIFOs are not implemented or not enabled, the data that is written to the RFWD is pushed into UART[1,0]x00[RBR].
- Read RBR register, it should be 0xaa, otherwise return fail

Like this for all the features of UART DMA IP, the test plan is developed and code has been developed using C language.

B. IIC Validation

The feature of IIC.

- IIC register access
- IIC Read in Standard Speed mode with 7-bit address
- IIC Write in Standard Speed mode with 7-bit address
- IIC Read in Fast Speed mode with 7-bit address
- IIC Write in Fast Speed mode with 7-bit address
- IIC Read in Fast Speed Plus mode with 7-bit address
- IIC Write in Fast Speed Plus mode with 7-bit address
- IIC Read in High Speed mode with 7-bit address
- IIC Write in High Speed mode with 7-bit address
- Bus Idle Condition Check
- Clock Stretching
- Transmission On Hold when Tx FIFO Has Data
- Flush Only I2C Tx FIFO During Transmit Abort
- Handling of NAK on Address
- Handling of NAK on Data
- General Call

As an example, the algorithm for one test case listed below.

- Send a start sequence
- Send the IIC address of the slave with the R/W bit low (even address)
- Send the location of internal register number you want to write to

- Send the data byte to be written
- Send the stop sequence to the slave

Similarly, for GPIO IP also, the test plan developed with detailed algorithm and code development is done.

IV.RESULTS

The ARM cortex SOC IPs: UART-DMA, IIC and GPIO are validated successfully. As an use case, we have taken ARM Cortex architecture based SOC, hence no silicon bugs found. These test cases if we run on new silicon, we may find bugs, so that semiconductor companies can confidently release the SOC to the customers.

V. CONCLUSION

A software diagnostic tool is developed successfully to validate Soc IPs, for UART-DMA, IIC and GPIO. To validate the selected IPs, we have chosen, ARM cortex architecture based processor, but the developed tool is not limited only for ARM, can also be used for other SoCs, to validate listed IPs. The developed diagnostic tool can also be extended to validate the other IPs with minimal effort. The tool we developed can be used to validate in pre-silicon stage or post silicon stage for new silicon. The best part of this project is, I have understood, the complete silicon validation process, which may fetch a better job in my career.

ACKNOWLEDGMENT

We thank to our project guide, Mrs.P.Siva Kalyani and, for providing necessary facilities towards carrying out this work. We are also very thankful to Mr. Samunuri Narayana raju, Embedded System Design Specialist, for all the support in successfully completing this project. We are also very much thankful to our project co-ordinator and entire ECE department faculty in giving the freedom in choosing this project, for continuous support and encouragement

REFERENCES

[1] Maik Boden, Jörg Schneider, Klaus Feske, SteffenRülke, "Enhanced Reusability for SoC-Based HW/SW Co-Design," DSD 2002, Pages: 94-101.

[2] Wolfram Hardt, "An Automated Approach to HW/SW-Codesign," partitioning in hardware-software codesign, IEE Collequim on 13 Feb. 1995 , Page(s):4/1 – 4/11.

[3] Massimo Baleani, Frank Gennari, Yunjian Jiang, Yatish Patel, Robert K. Brayton, Alberto Sangiovanni-Vincentelli, "HW/SW partitioning and code generation of embedded control applications on a reconfigurable architecture platform," International Conf,erence on Hardware Software Codesign, Proceedings of the tenth international symposium on Hardware/software codesign,Estes Park, Colorado , 2002, Page(s):151 –156.

[4] 7. S. Yoo, A.A. Jerraya, " Hardware/Software cosimulation from interface perspective," Computers and Digital Techniques, IEE Proceedings, Vol. 152.

[5] Xue Wang, Gang Yu, Jay Lee, "Wavelet Neural Network for Machining Performance Assessment and Its Implications to Machinery Prognostics," Proceedings of the 5th International Conference on Managing Innovations in Manufacturing (MIM)(2002).

[6] Guofeng Tong, Muammer Koc, Jay Lee, "System Performance Assessment Based on Control System Criteria Under Operating Conditions," Proceedings of the 5th International Conference on Managing Innovations in Manufacturing (MIM)(2002).

[7] Shui Yuan, Ming Ge, Hai Qiu, Jay Lee, Yangsheng Xu,"Intelligent Diagnosis in Electromechanical Operation Systems," Proceedings of the 2004 IEEE International Conference on Robotics & Automation New Orleans, LA, April 2004.

[8] Gang Yu, Hai Qiu, Dragan Djurdjanovic, Jay Lee, "Performance Prediction Using Recurrent Neural Network Modeling with Confidence Bounds,"International Conference on Intelligent Maintenance Systems, July 15-17, 2004, Arles, France.

[9] Wang Xue, Liu Chengliang, Jay Lee, "Intelligent Maintenance Based on Multi-sensor Data Fusion to Web-enabled Automation Systems," Proceedings of the Intelligent Maintenance Systems 2004, July 15-27, 2004-Arles, France

[10] YANG Kun, ZHANG Chun, DU Guoze, XIE Jiangxiang, WANG Zhihua, "A Hardware-Software Co-design for 11.264/AVG Decoder," E.E. Department, Tsinghua University Beijing, China,2006.

[11] W. Kayankit, W. Suntiamorntut, "A Hardware-Software Co-design for 11.264/AVG Decoder,"Proceedings of the 2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications, and Information Technology

BIOGRAPHY

Sri. LAHARI, studying M.Tech in Embedded Systems, at AITS, Rajampet, INDIA. Currently she is undergoing internship training at EmWare Technologies (INDIA) Pvt Ltd, at Bangalore center. She is expert in embedded systems peripheral drivers and 8/16/32 architectures etc.

Sri P. Siva Kalyani, Assistant Professor - He is having vast experience of Industry as well as Academics. He presented several papers in International and National Conferences and journals.