SCALABLE MULTICORE DESIGN FOR TEST INTERFACE DESIGN

CHONG SHI HOU

A project report submitted in partial fulfillment of the
requirements for the award of the degree of
Master of Engineering (Computer and Microelectronics System)

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

NOVEMBER 2007

To my beloved family

ACKNOWLEDGEMENTS

Throughout the period of doing this project, I received a lot of encouragements and assistance from my project supervisor, Professor Dr. Abu Khari b. A'ain. I would like to express my most sincere gratitude to him, for his guidance, support, motivation and help throughout this project.

I would also like to express my heartiest appreciation to my beloved mother and sister for their encouragement, love, understanding, care and ceaseless supports in all my endeavors.

Thirdly, I am very thankful to my department for sponsoring my master program and my managers, Patrick Tan Teck Wee and Chew Huat Chin, in particular, for their understanding and support for me to pursue this part-time Master study.

Finally, I would like to thank all my friends for their help and special thanks to University Technology Malaysia lecturers for sacrificing their weekends to travel to Penang for lecture.

ABSTRACT

With industries moving towards converged core design, and multicore processors products, DFx design and testing strategy need to be able to catch up with the pace of product development cycle, increase in test content and test time, as well as converged core design reuse in proliferation of products. As such, core-level test content must be reusable, multicore testing have to be done concurrently, while allow the choice of core isolation, as well as DFx multicore interface that are scalable to facilitate proliferation of products. The main objectives of this project are to investigate on major problems of multicore Design For Testability interface design, to analyze and understand pros and cons of multiple industrial multicore Design For Testability interface designs, to propose and to implement a novel design that can tackle three major problems in term of multicore Design For Testability interface design scalability, concurrent testability as well as trace reusability.

ABSTRAK

Memandangkan industri masa kini menuju kepada arah rekabentuk “converged-core” dan product-product pemproses berjenis multi-core, DFx sebagai antara satu strategi rekabentuk dan pengujian produk semikonductor menjadi semakin penting. Kini, fungsi DFx harus berselari dengan kemajuan dalam pembangunan produk yang semakin singkat, menampungi peningkatan test time dan test content, dan juga harus bersepadu dengan penggunaan semula rekabentuk “coverged-core”. Objektif utama project ini bertujuan untuk mengkaji dan menyelidik masalah-masalah yang wujud di rekabentuk antaramuka bagi multi-core “Design For Testability” (DFT), menganalisa dan memahami kelebihan serta keburukan aneka rekabentuk multicore DFT yang terdapat di industri masa kini. Di samping itu, hasil project ini juga membabitkan cadangan dan implementasi satu rekabentuk baru yang dapat bertujuan untuk mengatasi tiga masalah utama dalam rekabentuk multi-core DFT, iaitu “scalability”, “concurrent testability” dan “trace reusability”.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

API	–	Application Programming Interface
ATPG	–	Automated Test Pattern Generation
BIST	–	Built-in Self Test
DFT	–	Design For Test
DFX	–	Collective term for Design for Test (DFT), Design for Debug (DFD) and Design for Manufacturing (DFM)
DR	–	Data Register
FRC	–	Functional Redundancy Check
FSM	–	Finite State Machine
HVM	–	High Volumn Manufacturing
IP	–	Intellectual Property
IR	–	Instruction Register
IDCODE	–	Identification Code
MISR	–	Multiple Inputs Shift Register
MUX	–	Multiplexer
RTL	–	Register Transfer Language
SECT	–	Standard for Embedded Core Test
SoC	–	System-on-Chip
TAP	–	Test Access Port
TLM	–	TAP Linking Module
TTM	–	Time to Market

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

This project report proposes Scalable Multicore DFx Interface Design. The purpose of the design is to tackle various testing issues give rise from multicore designs and products. In this chapter, the issues and problem statements are discussed, providing a framework for the objectives of this project. This chapter covers the background and research motivation, problem statements, scope of work, constraints and assumptions, significant of the work, research methodology and finally the report organization.

1.1 Background and Research Motivation

Multicore processors are not a new invention in the 21st century. There is general consensus that the embedded market has been the leading innovator for architecting single-chip, multiprocessor systems. Since at least 1995 when the Texas Instrument (TI) TMS320C80 video processor was shipped, there have been off-the-shelf multicore CPUs on the market. Even prior to 1995, companies like Siemens, Phillips, Fujitsu, NEC, etc. might have built customized multicore CPU chips.

The exponential growth of cellphones, storage devices, consumer electronics, general purpose and server computing as well as automotive applications is driving the demand for multicore processing. Multicore processing is a growing industry trend as single core processors rapidly reach the physical limits of possible complexity and speed. Companies that have produced or are working on popular multicore products include Intel, AMD, ARM, Broadcom, Sun and IBM.

Homogeneous multicore products have been a common place in server computing as demonstrated by IBM, Sun, Intel and AMD over the years. Nevertheless, in recent years, it starts creeping into the desktop and mobile computing realms too. ARM MPCore Processor has also been offered for consumer devices from set-top boxes to cell phones.

Heterogeneous multi-core computing itself isn't particularly new. Such systems have been around since the mid-80 where a problem's workload is split between a general-purpose processor and one or more specialized, problem-specific processors. Notable historical examples include Floating Point Systems' array processors, the Inmos "Transputer" and the Connection Machine. Today's attached processor systems, besides GPUs, include ClearSpeed's accelerator systems and the Ageia PHYSX physics processing unit. In the processor realm, the IBM Cell Broadband Engine (a.k.a., "Cell BE" or simply, "Cell") is the best example of an entirely heterogeneous multi-core processor. The difference today is packaging: these processor systems are delivered as systems-on-a-chip (SOC). The heterogeneous multi-core SOC integration trend is very likely to continue in the future if IBM's Cell, the AMD/ATI merger or Intel in the GPGPU domain are indications of commercial trends.

Multicore architecture can increase efficiency of simultaneous processing of multiple tasks and can enable the designers to optimize computation and data flow with homogeneous or heterogeneous architectures. However, it also gives rise to the issues of duplicated front-end design efforts in converged core architectures, growing test contents in product development, increase in product development engineering headcounts, growing test time and tester platform costs, increase in the complexity of debugging multicore and intercore failures, etc.

Although dual-core and quad-core are just becoming a norm in recent months, the trend of increasing homogeneous or heterogeneous cores in microprocessor and SoC products will not stop here. As such, the research motivation in the area of multicore DFx interface design becomes clear – help to increase converged core design scalability, improve test content reusability, reduce test time, improve debugability and achieve significant test cost saving.

1.2. Problem Statements

With increase in the complexity of multicore processor design, such as more architectural features and more transistors per each physical core, as well as increase in the number of homogeneous and heterogeneous physical cores, such as dual-core, quad-core, eight-cores and even multi-heterogeneous cores SoC, many multicore related engineering issues have appeared, incurring duplicated design efforts, high number of product development headcounts, growth in the production test time, significant increment in the production test contents, additional tester costs, longer time-to-market, etc.

To be more technically specific, the issues can be categorized into the following problem statements.

(i) The problem of scalability of multicore DFx interface – Regardless of homogeneous multicore, heterogeneous multicore, multi-chip package products or multicore SoC, scalable multicore DFx interface has to be planned and designed upfront in the early stage of the project, otherwise the design will not be able to scale to more core, more hierarchical multi-chip packaging, or even reducing the cores for lower-end product segment. A non-scalable multicore DFx interface design will require a lot of design rework for product proliferations. The cost of redesign is huge in term of engineering resource, time and money.

(ii) The problems of trace reusability – Homogeneous or heterogeneous multicore products have their respective unique core-level traces. In many practical cases in the industries, these cores are not redesign from scratch, but they are rather instantiations of improvement from previous designs. As a result, high percentage of core-level and potentially chip-level trace reuse from their predecessors is expected. Low percentage of trace reuse can be directly translated into increase in test content volume, growing test time and engineering headcounts, increase in tester equipments, and as a result, significant increase in the production cost!

(iii) The problem of concurrent testability – For homogeneous multicore design, generally the cores are logically identical. Without the capability of concurrent testing, production testing in high volume manufacturing (HVM) will become multiple times of the single-core product. This issue will have serious impact to both the long test time as well as additional cost for more testers or testing platforms. In addition, without concurrent testing and comparing mode, engineers may need to go through iterations of pass-fail flow to determine the failing signature.

(iv) The problem of debugability – Without any multicore DFx interface capability for debug purpose, any core failures in the product will be tedious to debug as engineers may not have easy to use mechanism to quickly determine failing core and to further isolate the particular failing core. Without multicore DFx interface, more troublesome pass-fail flow needs to be used to determine the failing signature.

Based on limited time frame, as well as relative new research scope in this area, this research project is narrowed down to specifically focusing on the top three problems mentioned above, namely scalability, test content reusability and concurrent testing.

1.3. Objectives

After knowing the problem statements clearly, project objectives can be appropriately set as the following:

(i) The logical design of the multicore DFx interface must support upward lateral scaling, downward lateral scaling, as well as hierarchical scaling.

(ii) The design must also allow unique core test content as well as chip-level trace reuse up to certain significant extend, without substantially incurring any additional cost in terms of test time, test equipments and engineering resources.

(iii) Concurrent testing mechanism must also be supported to allow on-die comparison of test result for parallel testing, as well as the flexibility of choosing any healthy homogeneous core's signature as reference.

1.4. Scopes of Work

Regarding the above-mentioned objectives, the scope of work for this project will include:

(i) Performing architectural analysis of various open industry standard multicore DFx interface design, identify their pros and cons, and decide areas of improvement for a better multicore DFx interface architectural for design implementation.

(ii) Determining a parallel DFx feature (such as MISR) to be implemented together with Test Access Port (TAP) and the multicore DFx interface logics.

(iii) Determining a good choice of coding language. In this case, System Verilog is chosen over VHDL and Verilog95. In addition, determining a good choice of design simulator, in this case, Synopsys' VCS is chosen

over Altera's Quartus II and Mentor Graphics' Modelsim. This is explaint later in the thesis.

(iv) Implementing the prototype of the behavioral and gate-level logic designs of TAP FSM, performing logic simulation, and verifying the fundamental design correctness within the desirable functionalities.

(v) Researching and determining a good choice of pre-silicon validation language and tool, in this case low-level logic validation will use System Verilog language and VCS simulator, and high-level usage model validation will use e-Language and Specman, over the choice of user-defined Perl-Macro and user-defined API with VCS. Besides, effort will also be spent on proposing an appropriate combination of pre-silicon validation methodologies, flow and tools.

(vi) Researching and proposing test content reuse and concurrent testing strategies with multicore DFx design.

1.5. Constraints and Assumptions

The field of Multicore DFx Interface design can be very broad and complicated, especially when tens of serial and parallel DFx features are involved in the integration, or when complicated multiple frequency domains crossings are involved. In order to focus this project onto the Multicore DFx Interface design itself, the following list of constraints and assumptions are put in place to avoid the research and project from going astray.

(i) This project will focus on multiple homogeneous cores only.

(ii) All multicore DFx logics are operating in TCLK (TAP clock) domain. The TAP Clock and Core Clock are assumed to operate at a safe ratio, as such that the distribution delay of TAP signals in the core domain will not contribute to any speed path.

1.6. Significance of Work and Project Contributions

This project will contribute directly to the objectives mentioned above. By achieving the support for multicore DFx interface scalability, design cost will reduce significantly. In recent years, proliferation of multiple products from a converged core design is a common place. Any proliferation of the converged core design, such as increasing number of cores for high end market segment, say from 4 to 8, or reducing the number of cores for low end market segment, say from 2 to 1, will require lateral scalability of the multicore DFx interface support, without needing additional design resource or even significant redesign. As a result, the contribution with respect to this area can be as significant as allowing a very quick time to market (TTM) response for many proliferations or product line items deliveries, allowing a company to quickly respond to market demand, gaining various market segment shares or even responding aggressively to competitors' products with minimum design costs.

On the other hand, the capability of the multicore DFx interface design supporting test content reuse and parallel testing can be appreciated directly in term of the significant reduction in engineering resources of test content re-generation for many product proliferations of the same converged core design. It also allows significant saving in high volume manufacturing (HVM) test time and multi-million dollars of savings in functional, structural and system level tester platforms as well as product engineering headcounts.

For high volume microprocessors products such as general purpose CPUs and embedded cellphone processors, the savings contributed from the benefits of such multicore DFx interface design can easily be as much as a few hundred thousand dollars to tens of millions of dollars. As such, this importance to a company's operating cost and profit margin is undeniable.

1.7. Research Methodology, Techniques and Tools

In order to make the progress of this project smooth and achieve desirable objectives, a structured and realistic planning must be put in place. All working procedures, work loads and time lines shall be identified upfront. Time lines will be tracked separately using Gantt chart.

The initial stage of this project will be focusing on architectural research of multicore DFx interface designs, as well as tools and coding language research. Such research shall not take too long, yet they are very important in laying down the right foundation and pave the right way for subsequent project stages. Research will be done by reviewing various engineering journals and IEEE standards, as well as different vendor and in house tools and coding languages for design and validation. Any proprietary tools or languages will be avoided, to prevent unnecessary technical issues, difficulties in getting appropriate engineering support or even difficulties in portability of the design of this project.

For architectural research, four industry standards of multicore DFx interface designs that will be looked upon are IEEE P1500 Standard for Embedded Core Test, Whetsel's Multiple TAP Architecture, Oakland's Multiple TAP Architecture and Parulkar et al.'s Multiple TAP Architecture. Their respective designs and implementations will be scrutinized and analyzed, and conclusion will be drawn with respect to their pros and cons in terms of scalability, test content reusability and concurrent testability.

Next, modular design approach will be used for behavioral and logical designs. The designs will be sub-divided into multiple logical blocks, so that any prototyping codes can be written in corresponding design modules later. The design logics shall be modularized or broken down into blocks in such a way that each module are logically and functionally meaningful (such as TAP FSM), allow modular or unit level validation, has minimal interface connection with adjacent modules and has limited dependency on logical changes of other modular blocks.

These will ensure design progresses are smooth and systematic, minimize unnecessary changes and ease to manage design coding.

The multicore DFx interface logics will be designed first, follow by other logical blocks, such as Test Access Port (TAP) Finite State Machine (FSM), Boundary Scan feature, Instruction Registers, Bypass registers, etc. All the behavioral and logical designs will be analyzed with respect to their functionalities and usage models. Any potential design, implementation, simulation, validation and usage model challenges will also be discussed, together with potential mitigation plan proposals.

After the preliminary behavioral and logical design, research focus will be shifted towards pre-silicon validation domain, whereby various industry pre-silicon validation tools, methodologies and flows will be reviewed, analyzed. With that, a practical pre-silicon validation tools, methodologies and flows for this multicore DFx interface design will be proposed.

Apart from that, test content reusability and concurrent testability will also be focused from the perspective of production testing. Various testing platforms and usage models will be reviewed and discussed. Subsequently, a practical test content reuse and concurrent testing strategy will be proposed.

With the choice of System Verilog as the design coding language and Synopsys' VCS as the simulator, preliminary prototyping System Verilog coding will be attempted. Simulations will be run and results will be collected and analyzed. This section of the work can be tedious, and much iteration maybe needed, especially initial coding from scratch can be buggy. Coding, simulation, basic verification, debug and recoding cycles will be repeated until a fundamentally functional design implementations are produced. A lot of engineering effort and hours are expected for this part of the project. Once a functional prototype design has been produced, its corresponding simulation result will be discussed, together with potential mitigation plan proposed with respect to any challenges arise.

Last but not least, with lots of time and effort spent up to this stage of the project, a lot of conclusion can be drawn upon. Such technical experiences and implementation hardship will be very useful for proposing any future work for the continuity and improvements of this project.

1.8. Organization of Project Report

This report is organized into eight chapters. The first chapter is the introduction which covers the background, problem statements, objectives, scopes, the significant and contributions of the project. End of the chapter deals with the methodology, tools and techniques employed in this project.

Chapter 2 provides literature reviews of various industry multicore DFx interface architectures. Analysis will be made with respect to their scalability, test content reusability and concurrent testability. With that, detailed requirement for a better multicore DFx architecture design will be laid down, paving for the key milestone of this project.

Chapter 3 demonstrates real design and implementation of the multicore DFx interface architectures. It will start with the choice of modular design approach and selection of a good design language as well as a simulator to begin with; follow by high level block diagram and detailed logic implementation. Besides analysis the design itself, discuss will be done upon challenges encountered and corresponding mitigation plans.

Chapter 4 shows design and implementation with TAP (Test Access Port) Finite State Machine (FSM), Boundary Scan, as well as their integration with the multicore DFx interface. Similar to previous chapter, these design and implementation make user of modular approach, and the same choice of coding language and simulator. Analysis on design and discussion upon challenges encountered and corresponding mitigation plans will be covered too.

Chapter 5 shows simulation, verification and analysis of the simulated result. Simulation will be done to prove the success of implementation on dual-core serial mode, quad-core serial mode, dual-core parallel mode as well as quad-core parallel mode.

Chapter 6 briefly discusses the importance of pre-silicon validation and gives a literature review of various industry validation tools and methodologies. It then discusses the proposal of pre-silicon validation for this multicore DFx design. Chapter 7 then brings forward the importance of test content reusability and concurrent testability, literature review of their role in term of multicore testing requirements as well as the proposal for test content reuse and concurrent testing strategy.

Last but not least, final Chapter summarizes the works done as well as proposes future work for whoever intending to carry the research in similar scopes.