

# **REUSING DESIGN-FOR-DEBUG HARDWARE TO ENHANCE PERFORMANCE**

**Neetu Jindal**



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY DELHI

October 2019

©Indian Institute of Technology Delhi (IITD), New Delhi, 2019

# **REUSING DESIGN-FOR-DEBUG HARDWARE TO ENHANCE PERFORMANCE**

by

Neetu Jindal

Department of Computer Science and Engineering

Submitted

in fulfillment of the requirements of the degree of Doctor of Philosophy

to the



Indian Institute of Technology Delhi

October 2019

# Certificate

This is to certify that the thesis titled **Reusing Design-For-Debug Hardware to Enhance Performance** being submitted by **Miss Neetu Jindal** for the award of **Doctor of Philosophy** in Computer Science and Engineering is a record of bona fide work carried out by her under my guidance and supervision at the Department of Computer Science and Engineering, Indian Institute of Technology Delhi. The work presented in this thesis has not been submitted elsewhere, either in part or full, for the award of any other degree or diploma.

**Preeti Ranjan Panda**

Professor

Dept. of Computer Science & Engg.

Indian Institute of Technology Delhi

New Delhi- 110016

**Smruti Ranjan Sarangi**

Associate Professor

Dept. of Computer Science & Engg.

Indian Institute of Technology Delhi

New Delhi- 110016

To My Family

# Acknowledgements

After few years' frolic stint of Ph.D and while pushing the paddle to move to the next step of my research journey, this is high time to take a pause and pay my gratitude to all who supported me unconditionally. This journey would have never been so memorable and enjoyable without their support.

There is always a person in your life whose vision is a blueprint of your success. I would like to thank Dr. M.P. Gupta, who had assessed where I could excel and had visualized my career as a researcher. Like every engineer, I also got inclined towards civil services by the end of my B.Tech., and that was the moment when he had taken a stand against that magnificence and had shown me my true strengths and priorities. It was his dream that I am living today and I genuinely say that I am enjoying every moment of it. Thanks a lot Tau Ji, for all your unconditional love and support.

At the commencement of my Ph.D journey, every well-wisher had doubts on my research problem and suggested me to change my research path. It was Prof. Preeti Ranjan Panda who believed in me and supported me. He was always there to listen to me and encouraged me to fly as the sky is the limit. Today I laugh when I remember my absurd travel concern for internship in Noida and his critic reaction on my resistance. Now I travel almost every month and am always ready to travel to any part of the world without even thinking about it twice. Informal talks over weekly lunches had played a vital role in our professional as well as personal growth and became unforgettable lifetime memories. The way a pot-maker gives a perfect shape by

hitting from outside and supporting from inside, he has always been there to support me to achieve this milestone. Respect comes from the heart, but we need words to show that. I have only one single word to express all my gratitude to you i.e. “Sir”. Sir, you are a true mentor.

I express my heart-felt gratitude to my co-mentor Prof. Smruti Ranjan Sarangi for supporting me with his invaluable feedback and high-end computational infrastructure.

I would like to express my deep gratitude to Prof. Sanjiva for his patient guidance, enthusiastic encouragement and useful critiques in DHOOM research work.

I would like to express my very great appreciation to my Student Research Committee members - Prof. Balakrishnan, Prof. Sanjiva and Sharad Kumar for their valuable and constructive suggestions during the planning and development of this research work. I would like to thank Prof. Mausam and Prof. Anshul kumar for their useful suggestions during TA work. Their willingness to give me their time so generously has been very much appreciated. I would also like to thank Sharad Kumar and Deepak Chauhan from NXP Semiconductors, Noida for facilitating my visit to their office for internship and to learn real-world post-silicon validation techniques. I acknowledge the help provided by the IIT administrative and technical staff for their often underappreciated effort.

I would also like to acknowledge Shubhani and Divya for their efforts in developing a network-on-chip infrastructure in their course project, which was used in the experimental evaluation of the NoC work in this thesis.

I would also like to acknowledge efforts put in by Sandeep Chandran, Abhay Mitra, Kunal Singhal, Shubham Gupta and Shikhar Tuli at various stages of run-time verification research, which was used in the DHOOM work in this thesis.

This dissertation would not have been possible without a big fat army of amazing people, who supported me through thick and thin. Special thanks to - Sakshi Tiwari, Hadi Brais, Prathmesh Kallurkar, Priyanka Singla, Deepak Ravi, Ankit Anand, Ishani Mahajan, Saurabh Tripathi,

Palavi, Rajshekar Kalyappan, Solomon Abera, Dinesh Raghu, Dinesh Khandelwal, Divya Praneetha, Rajesh Kedia, Rahul Jain, Lokesh, Shubhankar, and Hameedah for always taking time from their schedule, no matter the day or the time, whenever I was in need. True friends are never apart, maybe in distance but never in heart.

A special thanks to the one stop to drain all our stress, Chaayos cafe, who used to make my special tea and was a reason for many to be annoyed.

At the end, I am especially grateful to my family, for supporting me emotionally and for being always there for me. I always knew that you believed in me and wanted the best for me. Thank you for teaching me that my job in life was to learn, to be happy, and to know and understand myself; because only then can I know and understand others. Thanks to my mother, Rekha Jindal, for guiding me, and for offering her generation expertise throughout this process. Young stars of my family, my nieces Parakshi, Charvi and Pihu, whose one innocent smile was more than sufficient to completely eliminate a whole bunch of strains.

**Neetu Jindal**

# Abstract

Decreasing feature sizes have caused ever increasing levels of on-chip component integration. The simulation or emulation used in pre-silicon validation can take a prohibitive amount of time to check for functional errors. During post-silicon validation, applications are executed on the chip prototype at native speeds and are analyzed using dedicated design-for-debug (DFD) hardware, enabling the discovery of functional bugs that may have slipped past pre-silicon validation. Post-silicon validation represents one of the most crucial and expensive components of the systems-on-chip validation methodology. It is performed under a highly aggressive schedule to satisfy time-to-market requirements. The DFD hardware is used to record the state history of important signals in the chip that could be critical in debugging the chip. The design of the DFD structure is non-trivial because it has to maximize the visibility while operating under very stringent area constraints, since it may become vestigial once the chip is in production. The simplest DFD structure is the trace buffer which consists of memory elements and records the values of signals deemed critical by the designer. These recorded signals can be extracted outside the chip and analyzed to determine the root cause of errors. With increasing complexity and higher levels of integration of modules on a single chip, the area consumed by DFD structures increases significantly. This is a challenge for the chip manufacturers because they have to strike a balance between the visibility and the area allocated to DFD structures. The two goals are competing; increasing visibility enables faster validation, but also increases the area overhead of the DFD hardware, which may become unusable when the chip goes into production (i.e., normal operation in field).

In this thesis, we address the above problem by repurposing the Design-for-Debug hardware through reusing it in-field to enhance the underlying architecture. Our novel design essentially leads to a win-win situation that is substantially advantageous for both validation and performance. This dissertation presents and evaluates three novel reusability scenarios of the validation hardware. In the first approach, we propose to reuse the core trace buffer as a victim cache that can be dynamically partitioned among multiple threads using machine learning algorithms. In the second approach, we propose a scheme AugVC to reuse trace buffers to augment the router buffer, with the objective of improving the overall network performance.

Further, we propose an architectural framework for runtime monitoring of program assertions, called DHOOM, which exploits the combination of a reconfigurable fabric present alongside a processor core, with the on-chip Design-for-Debug hardware. This combination of hardware features allows DHOOM to minimize the overall performance overhead of runtime verification, subject to a given area constraint. We present an algorithm for dynamically selecting an effective subset of assertion monitors that can be accommodated in the available programmable fabric, while instrumenting the remaining assertions in software.

# संक्षेप

ट्रांजिस्ट आकार के घटने के कारण ऑन-चिप घटक एकीकरण का स्तर बढ़ा है। प्री-सिलिकॉन सत्यापन में उपयोग किए गए सिमुलेशन या अनुकरण कार्यात्मक त्रुटियों की जांच करने के लिए एक निषेधात्मक मात्रा ले सकते हैं। सिलिकॉन के बाद के सत्यापन के दौरान, अनुप्रयोगों को मूल गति पर चिप प्रोटोटाइप पर निष्पादित किया जाता है और समर्पित डिज़ाइन-फॉर-डीबग हार्डवेयर का उपयोग करके विश्लेषण किया जाता है, जो उन कार्यात्मक त्रुटियों की खोज को सक्षम करता है जो पिछले प्री-सिलिकॉन में रह गए हों। पोस्ट-सिलिकॉन सत्यापन सिस्टम-ऑन-चिप सत्यापन पद्धति के सबसे महत्वपूर्ण और महंगे घटकों में से एक का प्रतिनिधित्व करता है। यह बाजार की आवश्यकताओं को पूरा करने के लिए एक अत्यधिक आक्रामक कार्यक्रम के तहत किया जाता है। डिज़ाइन-फॉर-डीबग हार्डवेयर का उपयोग चिप में महत्वपूर्ण संकेतों के इतिहास को रिकॉर्ड करने के लिए किया जाता है जो चिप को संशोधित करने में महत्वपूर्ण हो सकते हैं। डिज़ाइन-फॉर-डीबग हार्डवेयर की संरचना गैर-तुच्छ है क्योंकि इसमें बहुत कड़े क्षेत्र की बाधाओं के तहत काम करते समय दृश्यता को अधिकतम करना पड़ता है, क्योंकि चिप के उत्पादन के बाद यह अवशिष्ट हो सकता है। सरलतम डिज़ाइन-फॉर-डीबग हार्डवेयर ट्रेस बफर है जिसमें मेमोरी तत्व होते हैं और डिज़ाइनर द्वारा बताए गये महत्वपूर्ण संकेतों के मूल्यों को रिकॉर्ड करता है। इन रिकॉर्ड किए गए संकेतों को चिप के बाहर निकाला जा सकता है और त्रुटियों के मूल कारण को निर्धारित करने के लिए विश्लेषण किया जा सकता है। बढ़ती जटिलता और एकल चिप पर मॉड्यूल के एकीकरण के उच्च स्तर के साथ डिज़ाइन-फॉर-डीबग हार्डवेयर के क्षेत्र में काफी वृद्धि होती है। चिप निर्माताओं के लिए यह एक चुनौती है क्योंकि उन्हें दृश्यता और डिज़ाइन-फॉर-डीबग हार्डवेयर की संरचनाओं को आवंटित क्षेत्र के बीच एक संतुलन बनाना है। दो लक्ष्य प्रतिस्पर्धा कर रहे हैं; बढ़ती दृश्यता तेजी से सत्यापन को सक्षम करती है, लेकिन डिज़ाइन-फॉर-डीबग हार्डवेयर के क्षेत्र को भी बढ़ाती है, जो चिप के उत्पादन (यानी, क्षेत्र में सामान्य संचालन) में जाने पर अनुपयोगी हो सकता है।

इस शोध में, हम अंतर्निहित आर्किटेक्चर को बढ़ाने के लिए डिज़ाइन-फॉर-डीबग हार्डवेयर को इन-फील्ड में पुनः उपयोग के माध्यम से उपरोक्त समस्या का समाधान करते हैं। हमारा डिज़ाइन अनिवार्य रूप से एक जीत-जीत की स्थिति की ओर जाता है जो सत्यापन और प्रदर्शन दोनों के लिए काफी फायदेमंद है। यह शोध प्रबंध सत्यापन हार्डवेयर के तीन पुनः प्रयोज्य परिदृश्यों को प्रस्तुत करता है और उनका मूल्यांकन करता है। पहले दृष्टिकोण में, हम कोर ट्रेस बफर का पुनः उपयोग विक्टिम कॅश के रूप में प्रस्तुत करते हैं जिसे मशीन लर्निंग एल्गोरिदम का उपयोग करके कई थ्रेड्स के बीच गतिशील रूप से विभाजित किया जा सकता है। दूसरे दृष्टिकोण में, हम ट्रेस बफर्स को पुनः उपयोग में लाने के लिए एक योजना औगवीसी का प्रस्ताव करते हैं जिसमें राउटर बफर द्वारा समग्र नेटवर्क प्रदर्शन को बेहतर बनाते हैं।

इसके अलावा, हम कार्यक्रम के अभिकथन की निगरानी के लिए एक वास्तुशिल्प रूपरेखा का प्रस्ताव करते हैं, जिसे धूम कहा जाता है, जो एक साथ मौजूद पुनः प्रयोज्य चिप व डिज़ाइन-फॉर-डीबग हार्डवेयर के संयोजन का फायदा उठाता है। हार्डवेयर का यह संयोजन रनटाइम सत्यापन के समग्र प्रदर्शन के ओवरहेड को कम करने की अनुमति देता है, जो किसी दिए गए क्षेत्र की बाधा के अधीन है। हम सॉफ्टवेयर में शेष अभिक्रियाओं को लिखत करते समय उपलब्ध प्रोग्रामेबल फैब्रिक में समायोजित किए जा सकने वाले मॉनिटर के प्रभावी उपसमुच्चय का चयन करने के लिए एक एल्गोरिथम प्रस्तुत करते हैं।

# Contents

<b>Certificate</b>	<b>i</b>
<b>Acknowledgements</b>	<b>v</b>
<b>Abstract</b>	<b>ix</b>
<b>I Prologue</b>	<b>1</b>
<b>1 Introduction</b>	<b>3</b>
1.1 Background on Post-Silicon Validation . . . . .	6
1.1.1 Dedicated DFD hardware for validation . . . . .	8
1.1.2 Reusing Existing Architectural Components for Validation . . . . .	13
1.2 Thesis Contribution . . . . .	15
1.2.1 Reusing the core trace buffer as victim cache . . . . .	17
1.2.2 Reusing the trace buffer to augment router buffers . . . . .	19

---

1.2.3	Reusing the core trace buffer and reconfigurable fabric to support run-time verification . . . . .	22
1.3	Thesis Organization . . . . .	23
<b>II</b>	<b>Trace Buffer Reused as Victim Cache</b>	<b>25</b>
<b>2</b>	<b>Trace Buffer Reused as a Victim Cache</b>	<b>27</b>
2.1	Introduction . . . . .	27
2.2	Trace Buffer as Victim Cache . . . . .	29
2.2.1	Baseline architecture . . . . .	29
2.2.2	Proposed architecture . . . . .	31
2.3	Dynamic Power Optimization of Victim Cache . . . . .	35
2.4	Experiments . . . . .	38
2.4.1	Setup . . . . .	38
2.4.2	Impact of Trace Buffer Reuse . . . . .	39
2.4.3	Impact on Energy Delay Product . . . . .	42
2.4.4	Synthesis Results . . . . .	46
2.5	Chapter Summary . . . . .	46
<b>3</b>	<b>Dynamic Victim Cache Partitioning for SMT Cores</b>	<b>49</b>
3.1	Introduction . . . . .	49

---

3.1.1	Motivation . . . . .	50
3.2	High-level Architecture . . . . .	51
3.3	Multi-Mode Victim Cache . . . . .	53
3.4	Victim Cache Partitioning . . . . .	56
3.4.1	Victim Cache Partitioning Technique . . . . .	58
3.4.2	Data Collection Strategy for Training . . . . .	59
3.4.3	The Partitioning Controller . . . . .	60
3.5	Generalization to multiple threads . . . . .	61
3.6	Replacement policy . . . . .	64
3.7	Experiments . . . . .	66
3.7.1	Setup . . . . .	66
3.7.2	Impact of Victim Cache Partitioning . . . . .	66
3.7.3	Synthesis Results . . . . .	76
3.8	Chapter Summary . . . . .	76
 <b>III Trace Buffer Reused as Augmented Router Buffers</b>		<b>79</b>
 <b>4 Trace Buffer Augmented Router Architecture</b>		<b>81</b>
4.1	Introduction . . . . .	81
4.2	Background: Conventional Router Architecture . . . . .	85

4.3	Related Work . . . . .	87
4.4	Trace Buffer Augmented Router Architecture . . . . .	91
4.4.1	AugVC Router Architecture . . . . .	91
4.4.2	Output Port Directed VC Assignment (ODVC) . . . . .	95
4.5	Experimental Results . . . . .	99
4.5.1	Experimental Setup . . . . .	99
4.5.2	Synthetic workloads . . . . .	100
4.5.3	PARSEC and SPLASH2 Benchmarks . . . . .	110
4.5.4	Synthesis Results . . . . .	115
4.6	Chapter Summary . . . . .	117
<b>IV</b>	<b>Reusing Design-for-Debug Hardware for Online Monitoring</b>	<b>119</b>
<b>5</b>	<b>Design-for-Debug Hardware for Online Monitoring</b>	<b>121</b>
5.1	Introduction . . . . .	121
5.2	Related Work . . . . .	124
5.3	DHOOM Architecture . . . . .	125
5.4	Overview of DHOOM . . . . .	125
5.4.1	Architecture . . . . .	125
5.4.2	DHOOM Flow . . . . .	129

---

5.4.3	Design complexities . . . . .	131
5.5	Assertion mapping algorithm . . . . .	131
5.6	Experiments . . . . .	135
5.6.1	Setup . . . . .	135
5.6.2	Case Studies . . . . .	135
5.7	Conclusion . . . . .	139
<b>V</b>	<b>Epilogue</b>	<b>141</b>
<b>6</b>	<b>Conclusion and Future Work</b>	<b>143</b>
6.1	Summary of Contributions . . . . .	143
6.2	Future Research Directions . . . . .	145
	<b>Bibliography</b>	<b>149</b>
	<b>List of Publications</b>	<b>167</b>
	<b>Biography</b>	<b>169</b>



# List of Figures

1.1	Development lifecycle of system-on-chip. . . . .	4
1.2	Illustration of scan chain methodology. . . . .	7
1.3	Trace Buffers. . . . .	8
1.4	Architectural diagram of a system-on-chip with DFD hardware. The components highlighted in blue are the design-for-debug structures. . . . .	15
1.5	Reusing the core trace buffer as a victim cache. The modifications made to the conventional architecture to implement this design are highlighted in green. . .	18
1.6	Reusing the trace buffer present in the router to augment the router buffer. The modifications made to the conventional architecture to implement this design are highlighted in green. . . . .	20
1.7	Reusing the core trace buffer and reconfigurable fabric to support runtime verification. . . . .	22
2.1	High level illustration of the proposed approach. (a) Trace buffer used to store debug data. (b) Trace buffer reused as Victim Cache. . . . .	28
2.2	Baseline architecture of LEON3 . . . . .	30

2.3	Modified LEON3 architecture . . . . .	32
2.4	The index computation logic in the victim cache controller . . . . .	33
2.5	FSM of the victim cache controller. DC: data cache, VC: victim cache . . . . .	34
2.6	Top: Proposed modification for power optimization. Bottom: victim cache hits during a sequence of 70M instructions of the <i>perlbench</i> benchmark . . . . .	36
2.7	Power gating of the Victim cache. $N = 2$ . . . . .	37
2.8	Performance improvement for different data cache sizes . . . . .	40
2.9	Impact of varying trace buffer sizes . . . . .	41
2.10	Impact of varying associativity . . . . .	42
2.11	(a) Power Gating duration and (b) System EDP gain for different data cache sizes, $P=0.1$ and window size=400 . . . . .	43
2.12	Impact of victim cache power optimization for varying data cache sizes. Top: EDP gain without power opt. Bottom: EDP gain with power opt. $P=0.1$ , window size=400 . . . . .	45
3.1	Victim cache performance when <i>bzip2</i> and <i>gamess</i> are run concurrently . . . . .	50
3.2	High level architecture . . . . .	52
3.3	The victim cache mode selection process . . . . .	54
3.4	Multi-mode Victim Cache . . . . .	55
3.5	Victim Cache Partitioning . . . . .	56
3.6	Selection of class labels during training: alternating best and random selection . . . . .	59

---

3.7	Finite state machine for partition controller. class and curr_class refers to the number of ways assigned to thread0 for the corresponding class label . . . . .	60
3.8	Computation of the new victim cache assignment . . . . .	63
3.9	Replacement Policy . . . . .	65
3.10	Speedup comparison of multimode and Logistic Regression . . . . .	67
3.11	Performance Improvement using Multimode victim cache partitioning . . . . .	68
3.12	Performance Improvement using Logistic Regression . . . . .	70
3.13	Speedup comparison of partitioning using one-stage prediction and the FSM (Figure 3.7) . . . . .	72
3.14	Speedup comparison of partitioning using utility-based cache partitioning [1] and logistic regression) . . . . .	73
3.15	Performance Improvement using Logistic Regression for 4 threads and 8-way set associative victim cache . . . . .	75
4.1	Trace Buffer reused to augment router buffers. . . . .	84
4.2	Conventional Router Architecture. . . . .	86
4.3	AugVC Router Architecture . . . . .	90
4.4	Linked list of all the flits belonging to the same VC maintained using Next Address Table . . . . .	91
4.5	Output Port Directed VC Assignment (ODVC) . . . . .	96

4.6	Average packet latency under four synthetic traffics for different packet sizes (5, 10, 20, 30, 50, 100 flits). Dashed lines and solid lines represent Baseline5 and AugVC, respectively. . . . .	101
4.6	Average packet latency under four synthetic traffics for different packet sizes (5, 10, 20, 30, 50, 100 flits). Dashed lines and solid lines represent Baseline5 and AugVC, respectively. (cont.) . . . . .	102
4.7	Impact of different trace buffer sizes on average packet latency under uniform traffic (packet size = 100 flits) . . . . .	103
4.8	Impact of AugVC (Dashed lines) and ODVC (solid lines) design on average packet latency (normalized to Baseline5) . . . . .	104
4.8	Impact of AugVC (Dashed lines) and ODVC (solid lines) design on average packet latency (normalized to Baseline5) (cont.) . . . . .	105
4.9	Average packet latency (normalized to ODVC) variation of VOQ scheme under four synthetic traffics for different VC sizes (4, 5, 6, 7 and 8 flits per VC and packet size = 100 flits) . . . . .	107
4.9	Average packet latency (normalized to ODVC) variation of VOQ scheme under four synthetic traffics for different VC sizes (4, 5, 6, 7 and 8 flits per VC and packet size = 100 flits) (cont.) . . . . .	108
4.10	Normalized dynamic power (normalized to total power of Baseline5) under four synthetic traffics (injection rates = 0.24 for uniform, tornado and neighbor; 0.14 for transpose, packet size = 100 flits)). . . . .	109
4.11	Impact of varying trace buffer latencies and read holding buffer sizes. . . . .	111
4.12	Minimum, maximum and variance values of packet latencies for PARSEC abd SPLASH2 benchmarks . . . . .	112

---

4.13	Normalized average packet latency (normalised to Baseline5) for PARSEC and SPLASH2 benchmarks for three network sizes. . . . .	113
4.13	Normalized average packet latency (normalised to Baseline5) for PARSEC and SPLASH2 benchmarks for three network sizes. (cont.) . . . . .	114
5.1	High-level schematic of DHOOM. . . . .	123
5.2	Detailed architecture of DHOOM . . . . .	126
5.3	Monitoring system . . . . .	128
5.4	AssertMap Illustration . . . . .	133
5.5	Performance overhead versus hardware features . . . . .	136
5.6	Benefits of DHOOM . . . . .	137



# List of Tables

2.1	Synthesis Results . . . . .	46
3.1	Features Description . . . . .	57
3.2	Class labels and the corresponding partitioning . . . . .	57
3.3	An example of executing Algorithm 2 for 4 threads and 8 ways . . . . .	64
3.4	Area and power overhead for different controllers . . . . .	76
4.1	Proposed network configuration parameters . . . . .	98
4.2	Baseline configuration parameters . . . . .	99
4.3	Maximum injection rate of source processor with latency threshold and packet size of 1500 cycles and 100 flits, respectively. . . . .	110
4.4	Synthesis results normalized to Baseline5 . . . . .	115