# FPGA BASED CAM ARCHITECTURE STRING MATCHING FOR NETWORK INTRUSION DETECTION

GAN CHONG GIM

UNIVERSITI TEKNOLOGI MALAYSIA

# FPGA BASED CAM ARCHITECTURE STRING MATCHING FOR NETWORK INTRUSION DETECTION

GAN CHONG GIM

A project report submitted in partial fulfilment of the
requirements for the award of the degree of
Master of Engineering (Electrical - Computer and Microelectronic System)

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

JUNE 2012

*"To my beloved father, mother and brother, wife and son."*

## ACKNOWLEDGEMENT

In preparing this project report, I was in contact with many people, researchers, academicians, and practitioners. They have contributed towards my understanding and thoughts. In particular, I wish to express my sincere appreciation to my project supervisor, Dr. Muhammad Nadzir Marsono, for encouragement, guidance, critics and friendship. Without his continuing support and interest, this project report would not have been the same as presented here.

I am also indebted to Altera Corporation (M) Sdn. Bhd for funding my part-time Master study. My managers, UTM SPS, and FKE staffs also deserve special thanks for their assistance in providing stable and comfortable environment for me to focus on my research.

My fellow UTM SPS classmates should also be recognized for their support.

I am truly grateful to all my family members, who motivated me the most throughout this research. Without their support, I would not have gone so far.

*Gan Chong Gim*
*Penang - Malaysia*

## ABSTRACT

String matching for network processing is the method of analyzing if a particular pattern or signature is observed in the received packet or data. Executing string matching with software approaches could not meet multi-giga bandwidth specifications and very time consuming. A hardware string matching able to speed up the string matching process significantly. The focus of this project is to present hardware CAMs (Content Addressable Memories) based string matching to perform pattern searching process for network intrusion detection (NIDS) applications on Field Programmable Gate Array (FPGA). The hardware pattern matching system is designed and developed in Verilog RTL language targeting the Altera Stratix-III FPGA. The developed string matching system is simulated with Snort NIDS ruleset. Its results are evaluated in terms of the string matching delay and resource utilization. The algorithm is compatible to support flexible signature length and different number of signature sets requirements. The CAM based string matching architecture can be extended to support parallel signatures searching and approximate string matching.

# ABSTRAK

String yang sepadan untuk pemprosesan rangkaian merupakan satu kaedah menganalisis jika corak tertentu atau tandatangan diperhatikan dalam paket data atau diterima. Melaksanakan rentetan sepadan dengan pendekatan perisian tidak dapat memenuhi spesifikasi jalur lebar pelbagai Giga dan sangat memakan masa. Rentetan perkakasan sepadan dapat mempercepatkan string yang sepadan proses ketara. Fokus projek ini adalah untuk membentangkan CAMS (Content Addressable Memories) perkakasan rentetan berdasarkan padanan untuk melaksanakan proses mencari pola untuk pengesanan pencerobohan rangkaian permohonan Field Programmable Gate Array (FPGA). Sistem yang hampir sama pola perkakasan direka dan dibangunkan dalam bahasa RTL Verilog mensasarkan Altera Stratix-III FPGA. Rentetan sistem dibangunkan sepadan adalah simulasi dengan dengus NID Set Peraturan. Keputusan dinilai dari segi kelewatan string yang sepadan dan penggunaan sumber. Algoritma adalah serasi untuk menyokong panjang tandatangan fleksibel dan nombor yang berbeza keperluan set tandatangan. Rentetan CAM berasaskan sepadan seni bina boleh dikembangkan untuk menyokong tandatangan selari mencari dan pemadanan rentetan anggaran.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

## LIST OF ABBREVIATIONS

| | | |
|---|---|---|
| FPGA | - | Field Programmable Gate Array |
| NIDS | - | Network Intrusion Detection System |
| NIPS | - | Network Intrusion Prevention System |
| CAM | - | Content Addressable Memories |
| LDAP | - | Lightweight Directory Access Protocol |
| DNS | - | Domain Name System |
| DFA | - | Deterministic Finite Automation |
| NFA | - | Non-deterministic Finite Automation |
| LAB | - | Logical Array Block |
| ALM | - | Adaptive Logic Module |

# CHAPTER 1

# INTRODUCTION

Network intrusion detection is a method of detecting packet or data transmission across the computing world in order to detect abnormal or suspicious packet payload. This process involves the recognition of abnormalities in the packets transmission (anomaly based detection) or the identification of special signatures in the packets (signature-based detection). A rule-based network intrusion detection system (NIDS), also known as a signature-based NIDS, applies a number of patterns, or regulations, to outline characteristics of computing data that determine abnormal activities or specific attacks. The NIDS matches every transmitted data to each of the patterns, and asserted a signal when the data contains all of the characteristics identified by a regulation or rule. Protocol, destination port and address, data size, source address and port and packet content are the regular characteristics processed on the data.

## 1.1 Background and Research Motivation

Pattern matching is the method of analyzing if a particular pattern or signature is observed in the received packet or data. In computer network we have today, pattern matching is a very widely used process. For example, finding a desired file or looking for certain file contents to get desired text in computer files. Pattern matching can be partitioned into a few category. For example, exact pattern matching, as the name describes, searches for the exact string in the signature database. Approximate string matching, also known as pattern matching with errors, searches the nearest match on the patterns in the database [3]. There are many usages of pattern matching. Among them, the applications that use pattern matching the most are search engines and search areas. We normally will parse the database using a pattern as the keyword if we need to search any content from the search area. Of course, the key component in search engine is pattern matching.

The main challenge of string matching is the size of string ($P$) is far more than one ($P > 1$). Beside this, the incoming data or traffic ($T$) is not static. Using software approach is slow, and this presents bottleneck for meeting gigabit network applications. Recent executions of string matching with software approaches could not meet multi-giga bandwidth specifications and very time consuming [16]. A hardware string matching is able to speed up the string matching process significantly. Moreover, implementing this process in hardware enhances the analyzing time significantly and has some other benefits.

## 1.2 Problem Statements and Hypothesis

One of the well known algorithms to pattern matching is Content Addressable Memories (CAMs) [20, 18, 26] for fast matching of multiple signatures. Normal usages of CAM are in networking computations and caches or lookup tables. It is very common to use CAMs as the IP address lookup table in routers [5]. Both IP address lookup tables and caches are suitable applications for CAMs because their signatures are in fixed size – IP address for lookup tables, and address tag for caches.

We can easily execute the interpretation tables or memory array caches with software approach using various approaches including hash tables, binary trees, tries, etc. As computation bandwidth is getting faster and faster. It is more and more difficult for general purpose microprocessors or even specific network processors to meet the gigabit network requirements. Thus, hardware string matching approaches have come into picture, and for the most part, these approaches have been based on correspond or Content Addressable Memories (CAMs) [5, 23, 6].

The main disadvantage of CAMs base string matching is the huge memory area overhead, and this area overhead also brings to large power consumption. These disadvantages become an issue because area and power are important factors in hardware designs nowadays. Thus, we propose to apply discrete comparators [27, 12] to replace the memory array which consume very huge in area. It is expected that after we fixed the above mentioned disadvantages, our novel CAMs based string matching hardware should give high throughputs and low resource utilization.

## 1.3 Research Objectives

In view of the background and problem statement mentioned in previous slides, the aim of this project is to:

1. To design and develop a FPGA hardware CAM based string matching system. This is to improve the search time and enhance existing CAM based hardware string matching system in terms of string matching delay and resource utilization. We propose that, by enhancing the huge memory area overhead and power consumption tradeoff, the CAMs based hardware should be the optimum string matching algorithm in terms of hardware performance, power, and area.
2. To develop a hardware signature configuration compiler to automate the Verilog RTL modification whenever there is a signature update in NIDS rulesets. By having this design automation, it will not only ease the RTL coding conversion, it also shorten the turn around time to within a second and eliminate human error during the conversion process.
3. To analyze and simulate the developed FPGA hardware string matching system using Snort version 2.9.0.4 NIDS rulesets. Snort suites with different number of rules and flexible number of character lengths are the input to the hardware CAM base string matching system. The results are analyzed in terms of string matching performance and resource utilization, as well as comparison with previous works.

## 1.4 Scopes of Work

Based on available hardware and software resources and limited time frame, this research project is narrowed down to the following scope of work:

1. This project is not to design the entire network processor but only the string matching component.
2. This project is also not to develop new string matching algorithm, but to implement and enhance CAM based algorithms in hardware.
3. Evaluations and comparisons are carried out in terms of the performance and resource utilization.
4. The design is modeled using Verilog at the RTL abstraction level.

5. This project is limited to design, synthesis, simulate, place and route, and timing analysis using the Altera Quartus II design CAD.
6. The design is targeted for the Altera Stratix-III (EP3SL340) FPGA hardware.

## 1.5 Structure of the Report

The rest of this report is organized as follows. Chapter 2 provides a detailed literature reviews on string matching algorithms and their characterization. Pros and cons of each algorithm are also covered in the chapter. Chapter 3 introduces the proposed solutions for CAMs base string matching including its methodology, system and block level architectures, design details as well as the hardware signature configuration compiler. Chapter 4 presents the evaluation results of this project, in terms of performance and resource utilization. Chapter 5 is the conclusions and the proposed future works.

## REFERENCES


1. Field-programmable gate array. *http://en.wikipedia.org/wiki/FPGA*.
2. IEEE 802.3 Ethernet Working Group. *http://www.ieee802.org/3/*.
3. String searching algorithm. *http://en.wikipedia.org/wiki/String-matching*.
4. Alfred V. Aho and Margaret J. Corasick. Efficient String Matching: An Aid to Bibliographic Search. *Communications of the ACM*, 18(6):333–340, June 1975.
5. P. Francis. A.J. McAuley. Fast Routing Table Lookup using CAMs. *Proceedings of IEEE INFOCOM*, 3:1382–1391, 1993.
6. S.P. Khatri B. Gamache, Z. Pfeffer. A Fast Ternary CAM Design for IP Networking Applications. *Proceedings of International Conference on Computer Communications and Networks*, pages 434–439, 2003.
7. G. Milne B. Gunther and L. Narasimhan. Assesing Document Relevance with Run-time Reconfigurable Machines. *Proceedings of IEEE Symposium on FPGAs for Custom Computing Machines*, pages 10–17, Apr. 1996.
8. R.S. Boyer and J.S. Moore. A Fast String Searching Algorithm. *Communications of the ACM*, 20(10):66–72, Oct. 1977.
9. D. Schimmel C. Clark. Scalable Multi-pattern Matching on Highspeed Networks. *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines*, pages 249–257, 2004.
10. G. Davies and S. Bowsher. Algorithms for Pattern Matching. *Software-Practice and Experience*, 16(6):575–601, June 1986.
11. J.H. Morris D.E. Knuth and V.R Pratt. Fast Pattern Matching in Strings. *SIAM Journal on Computing*, 6(2):323–350, June 1977.
12. D. Pnevmatikatos I. Sourdis. Pre-decoded CAMs for Efficient and High-speed NIDS Pattern Matching. *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines*, pages 258–267, 2004.
13. Ronald P. Loui Michael Pachos James Moscola, John Lockwood. Implementation of a Content-Scanning Module for an Internet Firewall.

*Proceedings of IEEE Symposium on Field-Programmable Custom Computing Machines*, pages 31–38, Apr. 2003.

14. John A. Chandy Janardhan Singaraju. FPGA based String Matching for Network Processing Applications. *Microprocessors and Microsystems*, 32:210–222, 2008.

15. C.G. Sodini J.P. Wade. A Ternary Content Addressable Search Engine. *IEEE Journal for Solid State Circuits*, 24(4):1003–1013, 1989.

16. A. Dubois M. Boorman S. Poole V. Hogsett Granidt M. Gokhale, D. Dubois. Towards Gigabit Rate Network Intrusion Detection Technology. *Proceedings of International Conference on Field Programmable Logic and Applications,*, pages 404–413, 2002.

17. H. Nagai K. Takahashi. M. Hirata, H. Yamada. A Versatile Data String-search VLSI. *IEEE Journal for Solid State Circuits*, 23(2):329–335, 1988.

18. K. Hirata H. Ooka H. Yamada T. Enomoto M. Motomura, J. Toyoura. A 1.2-million Transistor, 33 MHz, 20-b Dictionary Search Processor (DISP) ULSI with a 160-kb CAM. *IEEE Journal for Solid State Circuits*, 25(5):1158–1164, 1990.

19. T. Enomoto M. Motomura, H. Yamada. A 2 k-word Dictionary Search Processor DISP with an Approximate Word Search Capability. *IEEE Journal for Solid State Circuits*, 27(6):883–891, 1992.

20. A. Mukhopadhyay. Hardware Algorithms for String Processing. *IEEE Computer*, pages 508–511, 1980.

21. D. Carver R. Franklin and B.L. Hutchings. Assisting Network Intrusion Detection with Reconfigurable Hardware. *Proceedings of IEEE Symposium on Field-Programmable Custom Computing Machines*, pages 111–120, Apr. 2002.

22. A. Mei R.P.S. Sidhu and V.K. Prasanna. String Matching on Multicontext FPGAs using Self-Reconfiguration. *Proceedings of ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pages 217–226, Feb. 1999.

23. Music Semiconductors. *Implementing an ARP Cache Using the MUAA CAM Family*. Tech. Rep. Application Note AN-N23, Princeton, NJ, February 1998.

24. R. Sidhu and V.K. Prasanna. Fast Regular Expression Matching using FPGAs. *Proceedings of IEEE Symposium on Field-Programmable Custom Computing Machines*, Apr. 2001.

25. Ioannis Sourdis and Dionisios Pnevmatikatos. Fast, Large-Scale String Match for a 10 Gbps FPGA-based Network Intrusion Detection System. *Proceedings of the 13th International Conference on Field Programmable Logic and Applications*, Sept. 2003.

26. A. Cantoni T. Moors. Cascading Content Addressable Memories. *IEEE Micro 3*, pages 56–66, 1992.

27. W.H. Mangione-Smith Y.H. Cho. Deep Packet Filter with Dedicated Logic and Read Only Memories. *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines*, pages 125–134, 2004.

28. Shiva Navab Young H. Cho and William H. Mangione-Smith. Specialized Hardware for Deep Network Packet Filtering. *Proceedings of 12th International Conference on Field Programmable Logic and Applications*, 2002.

29. V.K. Prasanna Z.K. Baker. A Methodology for Synthesis of Efficient Intrusion Detection Systems on FPGAs. *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines*, pages 135–144, 2004.

30. V.K. Prasanna Z.K. Baker. Automatic Synthesis of Efficient Intrusion Detection Systems on FPGAs. *IEEE Transactions on Dependable and Secure Computing*, 3(4):289–300, 2006.