IJETM

ISSN: 2394-6881

International Journal of Engineering Technology and Management (IJETM)
Available Online at www.ijetm.org
Volume 2: Issue 1: Page No. 42-45

An analysis of High-level graphical representation of the Crypto-Maniac system architecture

Jyoti Gupta [1], Nasreen Khan [2], Bhawna Yadav [3], Nidhi Sharma [4]

[1]M.Tech CSE, Vanasthali Vidyapeeth, Jaipur, Rajasthan

jyotigpt158@gmail.com

[2]Research Scholar, Department of computer science, Jaipur, Rajasthan

arajawat15@gmail.com

[3] Research Scholar, Department of computer science, Jaipur, Rajasthan

Yadav.7bhawna@gmail.com

[4]M.Tech Scholar, Rajasthan Institute of Engineering & Technology, Jaipur, Rajasthan

nids.bravo@gmail.com

ABSTRACT

The Crypto Maniac system architecture makes a strong case for high level application and specific architectures. The reason for its striking efficiency is simple an application specific design can achieve a level of efficiency that is impossible for general purpose architecture to obtain. In this paper, we present a specific the crypto Maniac system architecture specialized to efficiently execute cryptographic ciphers [1]. We carefully highlight the domain specific application characteristics identified and their accompanying optimizations in the Crypto Maniac system architecture. Detailed analyses of the design make a strong case for application specific optimization. In addition, our limited application domain creates opportunities to optimize the implementation, yielding superior performance. Our application specific design contains none of the baggage necessary to execute cryptographic workloads, making the specific resulting design.

Keywords: *High-level, Crypto, Architecture, Processing, Element, Optimization*

1. INTRODUCTION:

In order to reach better cipher performance, efficiency becomes the goal of our design. In pursuing our goal, the design will focus on a simple micro architecture, an efficient implementation of operations, and more efficient use of the clock cycle. Crypto Maniac is a 4-wide 32-bit Very Large Instruction Word machine with no cache and a simple branch predictor. Since kernel dependencies through registers and memory are well described, a static VLIW scheduler suffices. The lack of branch bottlenecks eliminates the need for a complex branch predictor [2]. A simple Branch Target Buffer (BTB) can correctly predict nearly all branches. We chose not to include a cache structure because code and data sets fit comfortably in a small static RAM. We employ a triadic (three input operands) ISA that permits combining of most cryptographic operation pairs for better clock cycle utilization. Finally, the Crypto Maniac processing elements can be combined into chip multiprocessor configurations for improved performance on workloads with inter-session and inter-packet parallelism.

2. Crypto Maniac system Architecture

The high level architecture of the Crypto Maniac processor show in figure 1. The host processor interfaces to the CM through the input (In Q) and output (Out Q) request queues. Cryptographic processing requests are inserted into the In Q by a host processor over a connecting bus [3]. The request scheduler distributes host processor requests, in the order received, to CM processing elements. The CM processing elements service requests from the In Q, placing any results produced in the Out Q for the host processor.

We envision that the input and output queues would be accessible by the host processor via a standard bus interface, such as a PCI bus [4]. It is sufficient to have one input queue for many CM processing elements, as the computationally intensive nature of cryptographic processing limits the bandwidth requirements on this interface.

The CM processing elements block waiting on a request from the host processor. When a request is dispatched to the CM processor, it uses to the request code to initiate the correct handler function [5]. When CM processing is complete, the tagged result of the computation is pushed into the Out Q for reception by the host processor. The host processor requests are tagged with a unique ID that can be used to identify requests as they complete and exit the Out Q. The unique ID permits requests to

*Corresponding author: Jyoti Gupta

complete out of order as may be the case with varied processing demands on CM processors. Also contained in the CM request is a session identifier that names a unique communication channel being processed in the crypto maniac [6].

The CM requests specify an operation for the CM to perform on the incoming data. Operations include:

Create a private key session, this request specifies the cryptographic algorithm, operating mode:

- Electronic codebook *vs*. chaining mode), and the private key of the session. Algorithm setup is performed during this request, creating key-specific substitution tables.
- Delete a private key session. This request releases all storage associated with a session.
- Encrypt/Decrypt data. The requested data is processed and the resulting cipher text or plaintext is returned in the result packet.

Additional administrative requests are supported that allow the host processor to initialize CM processor memory and redirect execution of individual CM processors [7].

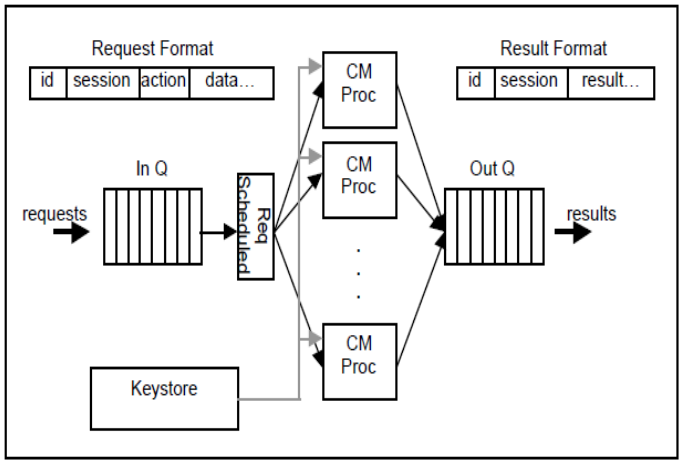

Figure 1: High-level Schematic of the Crypto Maniac Architecture

Requests arriving for CM processing are dispatched from the In Q to CM processing elements by the request scheduler. Requests are distributed first to a free CM processor. If multiple CM processors are free, the request is dispatched to the CM processing element that most recently processed a request in the same session. If there are no free CM processors in the same session, the request is assigned to the least recently used CM processor [8]. Directing requests to CM processors in the same session reduces the setup time to service the request, and when multiple CM processors are free but not in the same session, the least recently used CM processor is likely to contain an unneeded session context.

The key store is a high-density storage element that contains key-specific storage such as key data and substitution tables. The key store permits the Crypto Maniac to process simultaneous sessions on the same CM processor by storing key-specific data in the shared key store. When a new context is loaded into a CM processor, key specific data is transferred over a high performance interface to internal CM storage [9].

This data includes substitution data, permutation counters, and other internal algorithm state, at most 5k bytes for any of the algorithms implemented. Key setup is quite expensive for many algorithms, thus performance is greatly improved by having a convenient place to store key-specific data. The key store is an optional component to the Crypto Maniac design, it is only required when multiple sessions must be serviced simultaneously. In single session applications, such as virtual private networks and secure disk processing, the key store is not required.

3. Processing Element Architecture

The CM processing element is a simple 4-wide 4-stage VLIW processor as shown in Figure 2. Each cycle the pipeline fetches a single statically scheduled VLIW instruction word that contains four independent instructions. These four instructions access the register file in parallel while decoding. In the execute stage, instructions perform up to four parallel functional unit operations. In the write back stage of the pipeline (WB), the results of the CM instruction are written back to the register file.

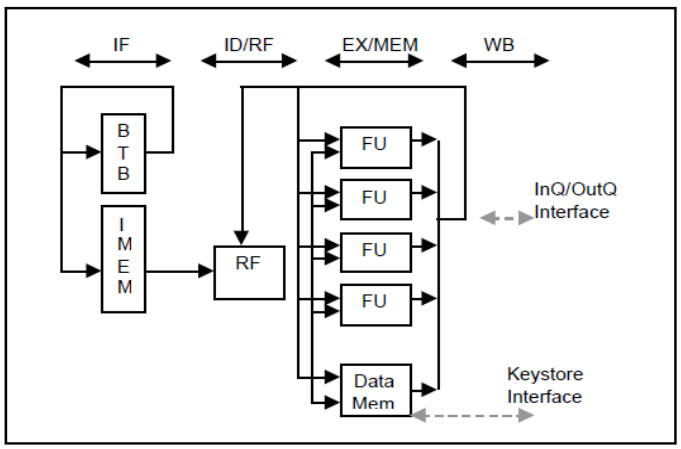

Figure 2: High-level Schematic of Crypto Maniac Processing Architecture

The front end of the CM pipeline contains a simple branch target buffer (BTB) used to predict branch targets.

The BTB contains the target address of a branch, any instruction that hits in the BTB is considered a taken branch. When a VLIW word contains multiple branches, the BTB always makes a prediction based on the last taken branch [10]. Most branches in cipher codes are trivial to predict as nearly all branches are taken branches at the end of cipher kernel loops. Instruction memory is accessed in parallel with the BTB, returning a VLIW instruction word at the end of the processor cycle.

Due to the small working set size of cipher algorithms, a very small BTB with 16 entries suffices. Moreover, the instruction memory need not be large. 1K bytes is sufficient to hold any of the cipher algorithms [11]. If key setup codes are kept off-chip, for example, by running setup codes on the host processor, many cipher kernels could be stored in a single 1K instruction memory.

4. Cryptographic Specific Optimizations

The Crypto Maniac crypto-processor is significantly more efficient than a general purpose microprocessor design [12]. In this section we list the specific program characteristics of cryptographic workloads that we identified, and the application specific optimizations that were applied to the Crypto Maniac that rendered performance, power, and area improvements. We order the optimizations from least to most aggressive.

i. Application Domain Characteristic:

Cryptographic workloads have well behaved data sets. Working set sizes are very small, and they exhibit high access locality.

ii. Application Specific Optimization: A simple, small memory system suffices. The Crypto Maniac contains a 512 byte instruction RAM, and a 4k byte data RAM. Caches or local DRAM is not required [13]. This results in memory system design with fast deterministic access latency, low power consumption, and an area efficient implementation.

5. CONCLUSIONS AND FUTURE SCOPE

We believe the Crypto Maniac design makes a strong case for application specific architectures. The reason for its striking efficiency is simple - an application specific processor design can achieve a level of efficiency that is impossible for general purpose designs to obtain. Our application specific design contains none of the baggage necessary to execute non-cryptographic workloads, making the resulting design smaller and cooler. In addition, our limited application domain creates opportunities to optimize the implementation, yielding superior performance. If general purpose processor optimization derails from the trajectory defined by Moore's Law, application specific architectures hold great potential to provide 1-2 generations of additional performance in the same technology, while giving even larger power and cost benefit.

6. REFERENCES

1. L. Wu, C. Weaver, and T. Austin. Crypto Maniac: A fast flexible architecture for secure communication. International Symposium on Computer Architecture Conference Proceedings, July 2001.
2. W.-C. Tsai, C.B. Shung, and S.-J. Wang. Two systolic architectures for modular multiplication. IEEE Transactions on VeryLarge Scale Integration (VLSI) Systems, 8(1):103-107, February 2000.
3. J. Burke, J. McDonald, and T. Austin. Architectural Support for Fast Symmetric-Key Cryptography. Proceedings of ASPLOS ,2000.
4. TILLICH,S.ANDHERBST, C. Boosting aes performance on a tiny processor core. In Proceedings of the Cryptopgraphers' Track at the RSA Conference on Topics in Cryptology (CT-RSA'08). vol. 4964, Springer, 170–186,2008.
5. Anderson, R., Bond, M., Clulow, J., and Skorobogatov, S. 2006. Cryptographic processors-a survey. Proc. IEEE 94, 2, 357--369.
6. Daniel J. Bernstein , Johannes Buchmann , Erik Dahmen, Post Quantum Cryptography, Springer Publishing Company, Incorporated, 2008.
7. Buchty, R., Heintze, N., and Oliva, D. Cryptonite A programmable crypto processor architecture for high-bandwidth applications. In Proceedings of the Organic and Pervasive Computing Conference (ARCS'04). vol. 2981, Springer, 184-198, 2004.
8. Lomonaco, M. 2004. Crypt array a scalable and reconfigurable architecture for cryptographic applications. Master's thesis, University of Central Florida.
9. R. Benadjila, O. Billet, S. Gueron, and M.J. Robshaw. The Intel AES Instructions Set and the SHA-3 Candidates. In ASIACRYPT '09: Proceedings of the 15th International Conference on the Theory and Application of Cryptology and Information Security, pages 162–178.Springer-Verlag, 2009.
10. J. Burke, J. McDonald, and T. Austin. Architectural support for fast symmetric-key cryptography. In Proceedings of the ninth international conference on Architectural support for programming languages and operating systems, pages 178–189. ACM, 2000.
11. D. Montgomery and A. Akoglu. Methodology and Toolset for ASIP Design and Development Targeting Cryptography-Based Applications. In Proceedings of IEEE International Conference on Application- Specific Systems, Architectures and Processors, pages 365–370, Montreal, Canada, July 2007.

12. Ajay K. Verma, Laura Pozzi, Paolo Ienne, Stefan Tillich, and Johann Grossed. When instruction set extensions change algorithm design: A study in elliptic curve cryptography. In4th Workshop on Application-Specific Processors (WASP 2005), page pp. 29, Jersey City, NJ, USA, September 2005

13. Theodoropoulos,D.,Siskos,A.,Andpnevmatikatos, D. . C proc: A custom vliw cryptography coprocessor for symmetric-key ciphers. In Proceedings of the 5th International Workshop on Applied Reconfigurable Computing (ARC'09). vol. 5453, Springer, 318–323, 2009.