An Analysis of Special Microprocessor Instructions from Intel, Texas Instruments, and Atmel Supporting AES and Other Cryptographic Algorithms

---

Final Design Specification

Shawn Wilkinson

ECE646

Fall 2015

Motivation and Description.

I am interested in comparing the performance of special microprocessor instruction sets using various hardware platforms. When Intel first announced their AES-NI instructions to help improve performance of AES calculations over software alone, numerous benchmarks were completed. Over the last five years, different hardware platforms have started incorporating their own hardware acceleration for AES but few comparisons between the two exist. I am interested in comparing the performance of an Intel i5 and i7 series chips with the use of the special AES instruction set, and then compare those results to the Texas Instruments AM3358 1GHz ARM Cortex-A8 with and without additional hardware designed to support cryptography operations. The ARMv8 instruction set contains new AES instructions similar to the AES-NI instructions released by Intel, while the Cortex-A8 contains special crypto hardware to improve performance. In addition to the hardware implemented in the Cortex-A8 processor, I would also like to examine additional hardware in the form of ASICs developed by Atmel to improve cryptography operation performance.

Platforms (hardware and software).

For hardware, I plan to use an Intel i7-4470k and i5-4670K processor on a traditional desktop PC, and a AM335x 1GHz ARM® Cortex-A8 onboard a Beaglebone Black embedded system, as well as the same Beaglebone Black system using a CryptoCape, which comes with several Atmel integrated circuits designed for enhancing several cryptography operations. For software, I plan on using GCC to compile readily available open-source C code within a Linux environment.

Libraries.

I plan on using the wolfCrypt library from wolfSSL. It is an open-source cryptography library developed in C and Assembly. I also plan on using the SUPERCOP cryptography benchmarking software created by VAMPIRE.

Assumptions: Optimization targets, limits on memory, etc.

The hardware I'm using will have more than enough memory to handle the tasks I will ask it to perform, and do not expect to have to optimize code to work around hardware limitations.

Detailed specification of input and output of the program, including exact format of input/output files.

I plan to encrypt and decrypt file sizes, both plaintext ASCII and binary, from very small to very large.

References to detailed descriptions of implemented functions.

I will implement AES based on the formal standard published by the National Institute of Standards and Technology in Federal Information Processing Standards Publication 197, published November 26, 2001. Modes of operation that will be tested include CBC, CTR, GCM, and CCM.

Initial analysis of similar reports.

My references include several benchmarks of the AES-NI instruction set when it first became available. Most of the benchmarks simply used the benchmark program available with TrueCrypt software to compare the performance of AES encryption with the AES-NI instruction set enabled and then without. I plan on using SUPERCOP (System for Unified Performance Evaluation Related to Cryptographic Operations and Primitives) to run my own benchmarks, and will be able to compare to other reports previously published by users of this software.

Procedures for testing functionality and performance.

I will encrypt and decrypt files using the SUPERCOP benchmarking software. The programs which run the encryption and decryption will keep timers and output the exact amount of time the entire operation required using various modes of encryption. Verification of successful encryption can be completed using a known third party decryption algorithm or program.

Plan of experiments to be performed.

I would like to test the speed of encryption and decryption for the following hardware/software configurations, using the various block modes of AES (CBC, CTR, GCM, and CCM):

Intel i5 Chip: AES-NI disabled
Intel i5 Chip: AES-NI enabled
Intel i7 Chip: AES-NI disabled
Intel i7 Chip: AES-NI enabled
TI ARM: Crypto hardware disabled
TI ARM: Crypto hardware enabled
TI ARM: Using Atmel ASICs

Time schedule.

Oct 26-27: All software is installed on all platforms
Nov 9-10: Benchmarks are able to run on all platforms
Nov 23-24: Completion of all tests and analysis of Results
Dec 7-8: Final Report and Presentation Completed

Possible changes depending on progress of project.

If given time, I would also like to further analyze performance increases by using special ICs on the Beaglebone Black CryptoCape, such as the Atmel ATAES132 chip, which performs AES-128 encryption, and the Atmel ATSHA204 chip, which performs SHA-256 hash algorithms. I would like to talk about

additional benefits to special instruction sets besides performance, any disadvantages to using special instruction sets, and how the hardware accelerators and ASICs used to enhance performance operate.

Tentative TOC for final report.

Introduction
Hardware and Software Specifications
Details on Benchmarking Algorithm
Testing results of Intel i7 and i5 chip
Testing results of ARMv8 chip
Testing results of Atmel ICs supporting ARMv8 chip
Analysis and Comparison of Results
Future Tests

List of literature.

[1] AES Instruction Set. (n.d.). *Wikipedia*. Available: https://en.wikipedia.org/wiki/AES_instruction_set. Accessed Sep. 22, 2015.

[2] S. Gueron. (2009). Intel's New AES Instructions for Enhanced Performance and Security. LNCS, vol. 5665, pp. 51-66. Available: https://www.iacr.org/archive/fse2009/56650054/56650054.pdf

[3] J. Bos. (2011) Efficient hashing using the AES instruction set. LNCS, vol. 6917, pp. 507-522. Available: http://eprint.iacr.org/2010/576.pdf

[4] S. Gueron, "AES-GCM software performance on the current high end CPUs as a performance baseline for CAESAR competition," presented at DIAC Conf., Chicago, IL, 2013. Available: http://2013.diac.cr.yp.to/slides/gueron.pdf

[5] R. Benadjila, "Use of the AES instruction set," presented at Ecrypt II Conf., Bruges, Belgium, 2012. Available: https://www.cosic.esat.kuleuven.be/ecrypt/AESday/slides/Use_of_the_AES_Instruction_Set.pdf

[6] How exactly does AES-NI work?. (2014, Oct.). Stack Exchange. [Online]. Available: http://crypto.stackexchange.com/questions/19544/how-exactly-does-aes-ni-work

[7] ECHO Hash function. (n.d.). France Telecom. [Online]. Available: http://maketheconnection.net/conditions/ptsd2. Accessed Sep. 22, 2015.

[8] ARMv8 instruction set overview. (2011, Nov. 11). ARM Limited. [Online]. Available: http://www.element14.com/community/servlet/JiveServlet/previewBody/41836-102-1229511/ARM.Reference_Manual.pdf

[9] G. Hofemeier. (2012, Jul. 26). Introduction to Intel AES-NI and Intel Secure Key Instructions. [Online]. Available: https://software.intel.com/en-us/articles/introduction-to-intel-aes-ni-and-intel-secure-keyinstructions

[10] Some small suggestions for the Intel instruction set. (2014, May 17). The cr.yp.to blog. [Online]. Available: http://blog.cr.yp.to/20140517-insns.html

[11] D. Bernstein and T. Lange (editors). "eBACS: ECRYPT Benchmarking of Cryptographic Systems. http://bench.cr.yp.to. Accessed Oct. 15, 2015.