ReadMe

May, 2011

Version 1.2

Intel® AES New Instructions (AES-NI) Sample Library

Overview

The Intel® AES New Instructions (AES-NI) Sample Library demonstrates how one might implement a high performance Advanced Encryption Standard (AES) block cipher using the new AES-NI instructions available in Intel Core™ i5, i7, Xeon® 5600 series and newer processors. All code samples can be compiled and run as native 32-bit or 64-bit binaries for both Microsoft® Windows® and Linux operating systems. This package is divided into three parts; The Intel® AES sample library, an AES example implementation using the library, and an application that compares Dr. Brian Gladman's AES performance with that of the AES-NI optimized library.

Intel® AES sample library

The Intel® AES sample library demonstrates how one might implement an optimized AES block cipher using AES-NI. The library supports the required 128, 192, and 256 bit key sizes along with the AES-CBC, and AES-CTR cipher modes. The library has not been extensively validated and therefore is not intended for production usage.

The `intel_aes_lib` directory contains the library source code along with a verification and timing application (Windows® only). The sources are split into two types. Assembler files consisting of the optimized AES implementation and timing utility can be found in the `asm` directory. C/C++ files consisting of the verification app and sample timing app can be found in the `src` directory. File descriptions are below:

`intel_aes_lib/src/intel_aes.c` - C wrapper for AES library assembler routines.

`intel_aes_lib/src/aessample.c` - Small verification app which demonstrates how to call the AES library routines and executes the NIST test vectors for each mode and key size.

`intel_aes_lib/src/aessampletiming.cpp` – Application to generate AES encrypt /decrypt sample timings (cycles/byte) using a multitude of factors like; key size, buffer size, thread count, mode, etc. This application only supports Windows®.

`intel_aes_lib/asm/x86/do_rdtsc.s` – 32-bit read time stamp counter module used for performance timings.

`intel_aes_lib/asm/x86/iaesx86.s` – 32-bit assembler implementation of AES-NI optimized AES routines.

`intel_aes_lib/asm/x64/do_rdtsc.s` – 64-bit read time stand counter module used for performance timings.

`intel_aes_lib/asm/x64/iaesx64.s` - 64-bit assembler implementation of AES-NI optimized AES routines.

`intel_aes_lib/where_files_come_from_and_license.txt` - Details regarding the software license.

AES Example Application

The AES Example is based off the original tutorial presented by Laurent Haan which can be found here. The intent of the tutorial is to show how to implement the AES cipher in a way that is easy to understand, and is not meant to be a performance test. Modifications to the tutorial code have been made to incorporate the usage of the Intel® AES sample library which provides a SIGNIFICANT speedup. The AES Example supports the use of 128, 192, and 256 bit keys for use with straight AES block encryption/decryption, AES-CBC, and AES-CTR modes. For more information on the available command line options be sure to use the "**-h**" option. AES Example sources can be found in the **aes_example** directory. Details regarding the software license for the sources can be found in the file **aes_example\where_files_come_from_and_license.txt**.

AES Gladman Subset

Dr. Brian Gladman's AES assembly implementation has been quoted in various published performance papers as being some of the most efficient AES assembler code available and can be found here. The intent of the **modetest** application is to compare Dr. Gladman's high performance software implementation with that of the AES-NI optimized Intel® AES sample library. The source file **modetest.c** has been modified to use the Intel® AES sample library so that a comparison can be made with regard to cycle/byte efficiency of the two implementations. The **modetest** sources can be found in the **aes_gladman_subset/src** directory. The application supports modifications to the number of blocks processed, crypt method, number of threads, CPU affinity mask, key size, number of loops, and the length of time to run. Both the Gladman and Intel® AES sample library routines support 128, 192, and 256 bit key sizes as well as AES-CBC and AES-CTR modes. For more information on the available command line options be sure to use the "**-h**" option. . Details regarding the software license for the sources can be found in the file **aes_gladman_subset/where_did_gladman_code_come_from_and_license.txt**.

Supported Development Environments and Tools

Microsoft® Visual Studio® 2005

Microsoft® Visual Studio® 2008

Microsoft® Visual Studio® 2010

GNU C compiler 4.x

YASM v0.8 (or newer) assembler

Microsoft® Windows® Build Instructions

For 32-bit Windows®, execute the following command line from within a "Visual Studio Command Prompt" shell:

```
mk_win86_all.bat
```

For 64-bit Windows®, execute the following command line from within a "Visual Studio x64 Win64 Command Prompt" shell:

```
mk_win64_all.bat
```

Linux* Build Instructions

For 32-bit Linux, execute the following shell script on the command line:

```
./mk_lnx86_all.bat
```

For 64-bit Linux, execute the following shell script at command line:

```
./mk_lnx64_all.bat
```

Fixed Issues and Updates in this build

- Resolved an issue in the 32-bit ASM implementation of AES-CTR-[192|256] where a key check was jumping to the wrong assembler label.

v1.1

- Resolved counter 'endian-ness' issue in ASM implementation of AES-CTR mode in the Intel® AES sample library.
- Revised '**`aessample[86|64].exe`**' to use NIST test vectors for AES-CBC and AES-CTR which are defined in NIST Special Publication 800-38A (Appendix F).

License Information

- License information and details vary per source file. Be sure to check the individual source files and "where_files_come_from_and_license.txt" for licensing details.

Notices

Intel sales office or your distributor to obtain the latest specifications and before placing your product order. Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or by visiting Intel's Web Site at http://www.intel.com/.