ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 3, May 2014

Parallel AES Encryption with Modified Mix-columns For Many Core Processor Arrays

M.S.Arun, V.Saminathan

Abstract*— AES is an encryption algorithm which can be easily implemented on fine grain many core systems. It is a symmetric algorithm and takes a 128 bit data block as input and performs four rounds of transformations to generate output ciphertext.The Asynchronous Array of simple Processors is an example of fine grained many core systems. Each implementation exploits different levels of data and task parallelism.AES is implemented on the Asynchronous Array of simple Processors with online key expansion. Offline Key expansion can be done by removing the cores used for Key expansion from online designs. AES implementations on One-Task One-Processor, Loop-Unrolled Nine Times and Loop-Unrolled Three Times are done. AES is also implemented on Hardware.

***Index Terms*— Asynchronous Array of simple Processors (AasP), Advanced Encryption Standard (AES), One-Task One-Processor (OTOP)**

I. INTRODUCTION

This paper presents various software implementations of the AES algorithm with different data and task parallelism granularity, and shows that AES implementations on a fine-grained many-core system can achieve high performance, throughput per unit of chip area and energy efficiency compared to other software platforms. Both the online and offline key expansion process for each implementation model are discussed. This paper compares the area efficiency among different implementations.Final session concludes the paper.

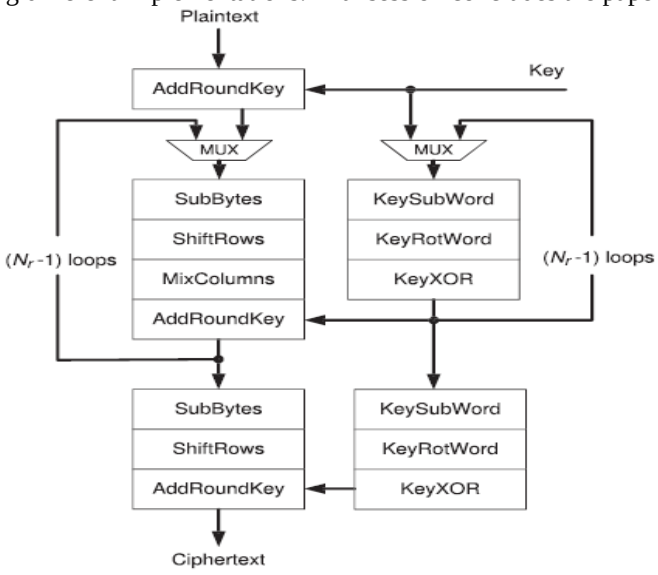


Fig. 1. Block diagram of AES encryption

II. ADVANCED ENCRYPTION STANDARD

AES is a symmetric encryption algorithm, and it takes a 128-bit data block as input and performs several rounds of transformations to generate output cipher text. Each 128-bit data block is processed in a 4-by-4 array of bytes, called the state. The round key size can be 128, 192 or 256 bits. The number of rounds repeated in the AES, N_r, is defined by the length of the round key, which is 10, 12 or 14 for key lengths of 128, 192 or 256 bits, respectively. Fig. 1 shows the AES encryption steps with the key expansion process. For encryption, there are four basic transformations applied as follows:

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 3, May 2014

1. **SubBytes**: The SubBytes operation is a nonlinear byte substitution. Each byte from the input state is replaced by another byte according to the substitution box (called the S-box).

2. **Shift Rows**: In the Shift Rows transformation, the first row of the state array remains unchanged. The bytes in the second, third, and forth rows are cyclically shifted by one, two, and three bytes to the left, respectively.

3. **Mix Columns**: During the Mix Columns process, each of the state array is considered as a polynomial the result is the corresponding column of the output state.

4. **AddRoundKey**: A round key is added to the state array using a bitwise exclusive-or (XOR) operation. Round keys are calculated in the key expansion process. If Round keys are calculated on the fly for each data block, it is called AES with online key expansion. On the other hand, for most applications, the encryption keys do not change as frequently as data. As a result, round keys can be calculated before the encryption process, and kept constant for a period of time in local memory or registers. This is called AES with offline key expansion. In this paper, both the online and offline key expansion AES algorithms are examined.

Similarly, there are three steps in each key expansion round.

1. **KeySubWord**: The KeySubWord operation takes a four-byte input word and produces an output word by substituting each byte in the input to another byte according to the S-box.
2. **KeyRotWord**: The function KeyRotWord takes a word performs a cyclic permutation, and returns the word as output.
3. **KeyXOR**: Every word is equal to the XOR of the previous word, and the word Nk positions earlier.

The decryption algorithm applies the inverse transformations in the same manner as the encipherment. As a result, we only consider the encryption algorithm in this work for simplicity, since the decipherment yields very similar results.

III. TARGETED MANY-CORE ARCHITECTURE

Fine-Grained Many-Core Architecture -According to Pollack's Rule, the performance increase of architecture is roughly proportional to the square root of its increase in complexity. The rule implies that if we double the logic area in a processor, the performance of the core speeds up around 40 percent. On the other hand, a many core architecture has the potential to provide near linear performance improvement with complexity. For instance, instead of building a complicated core twice as large as before, a processor containing two cores (each is identical to the other) could achieve a possible 2_ performance improvement if the application can be fully parallelized. Therefore, if the target application has enough inherent parallelism, architecture with thousands of small cores would offer a better performance than one with a few large cores within the same die area.

One-Task One-Processor (OTOP)

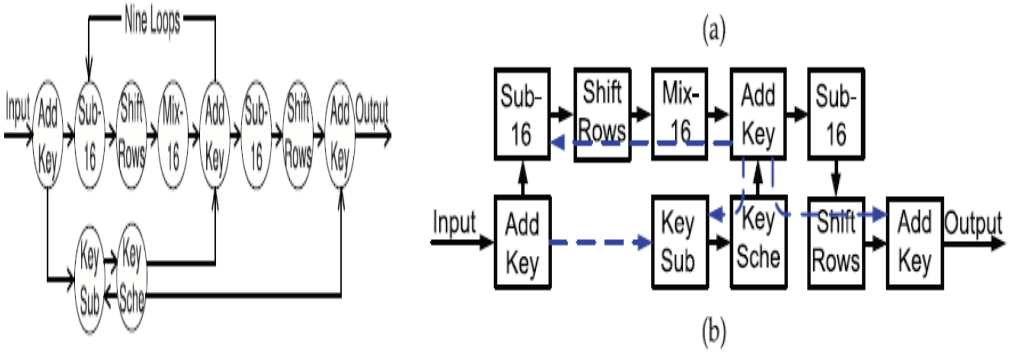


Fig. 2. One-Task One-Processor (OTOP)

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 3, May 2014

The most straightforward implementation of an AES cipher is to apply each step in the algorithm as a task in the dataflow diagram as shown in Figure 2. Then, each task in the dataflow diagram can be mapped on one processor on the targeted many-core platform. We call this implementation one-task one-processor. For simplicity, all of the execution delay input rates, and output rates in the following dataflow diagrams are omitted. Since the key expansion is processing in parallel with the main algorithm, the throughput of the OTOP implementation is determined by the nine (N_r _ 1 ¼ 9) loops in the algorithm. The OTOP implementation requires 10 cores on AsAP.

Loop-Unrolled Nine Times

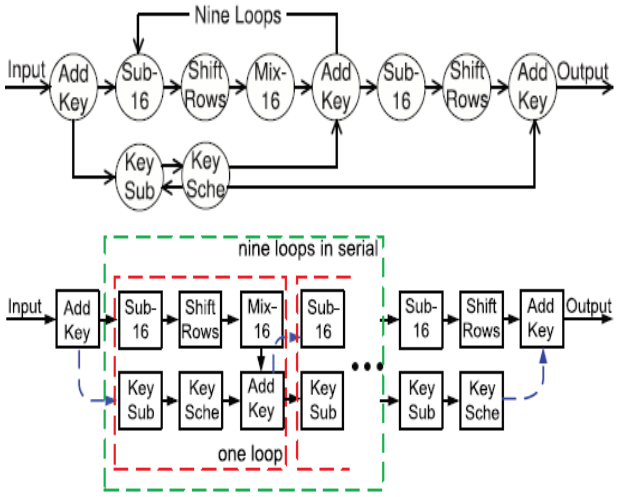


Fig. 3. Loop-Unrolled Nine Times

To enhance the AES cipher's throughput, we apply loop unrolling to the OTOP model and obtain the Loop-unrolled Nine Times dataflow diagram as shown above. The loop unrolling breaks the dependency among different loops and allows the nine loops in the AES algorithm to operate on multiple data blocks simultaneously. To improve the throughput as much as possible, we unroll the loops in both the AES algorithm and the key expansion process by N_r _ 1 and N_r times, which equals 9 and 10, respectively. After loop unrolling, the throughput of the AES implementation is increased to 266 cycles per data block, equaling 16.625 cycles per byte. The mapping of the Loop-unrolled Nine Times model is shown in, which requires 60 cores.

Loop-Unrolled Three Times

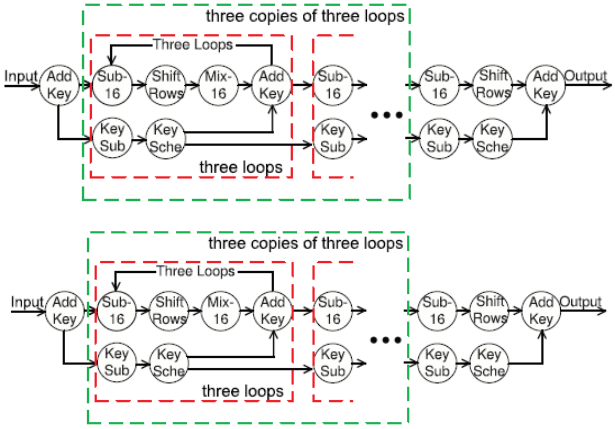


Fig. 4. Loop-Unrolled Three Times

To achieve a moderate throughput with fewer cores, we could unroll the main loops in the AES algorithm by S times (S is divisible by N_r _ 1), instead of N_r _ 1 times. For this example, the nine loops in the AES algorithm could be split into three blocks, and each block loops three times. The dataflow diagram and mapping of the Loop-unrolled three Times implementation are shown in Figure 4. Compared to the OTOP model, the throughput is improved to 1,098 cycles per data block, which equals 68.625 cycles per byte; while the mapping requires 24

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 3, May 2014

cores, 36 fewer than the Loop-unrolled Nine Times implementation.

Parallel-Mix Columns

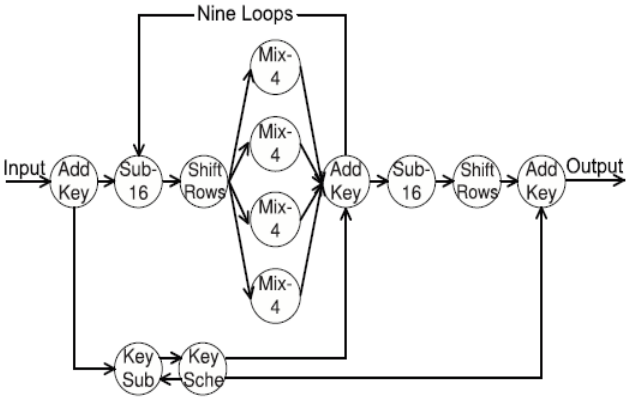


Fig. 5.1 Parallel-Mix Columns

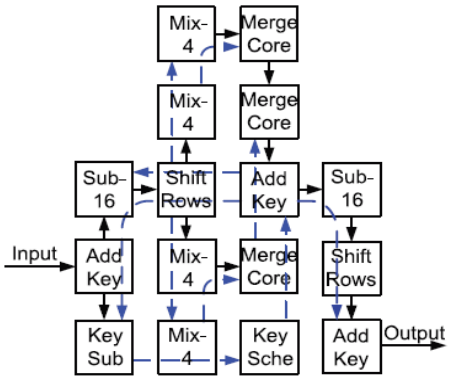


Fig. 5.2 Parallel-Mix Columns

Besides loop unrolling, another way to increase the throughput of the OTOP model is to reduce the main loop's latency in the AES algorithm. In a single loop, the execution delay of MixColumns-16 results in 60 percent of the total latency. Each MixColumns-16 operates on a four-column data block, and the operation on each column is independent. Therefore, each MixColumns-16 processor can be replaced by four MixColumns-4s. Each MixColumns-4 actor computes only one column rather than a whole data block. As a result, the throughput of the Parallel-Mix Columns implementation is increased to 2,180 cycles per block, equaling 136.25 cycles per byte. The dataflow diagram and mapping of the Parallel-Mix Columns model are shown in Fig. 5

Parallel-SubBytes-MixColumns

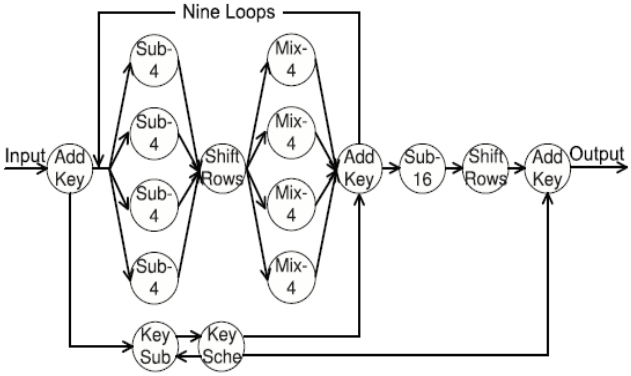


Fig. 6.1 Parallel-Sub Bytes-Mix Columns

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 3, May 2014

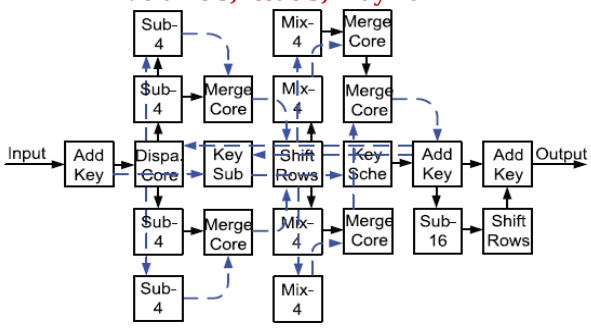


Fig. 6.2 Parallel-Sub Bytes-Mix Columns

In the Parallel-MixColumns implementation, SubBytes-16 requires 132 cycles to encrypt one data block, which contributes the largest execution delay in one loop. In order to increase the throughput further, we parallelize one SubBytes-16 into four SubBytes-4s, which is shown in Fig. 6. In this implementation, each SubBytes-4 processes 4 bytes rather than 16 bytes in one data block. The effective execution delay of the SubBytes process is decreased to 40 cycles per block, only around one fourth as before. Therefore, the throughput of the Parallel-SubBytes-Mix Columns model is increased to 1,350 cycles per block, equaling 84.375 cycles per byte. The mapping graph of the Parallel-SubBytes-Mix Columns implementation on AsAP shown in Fig. 6 requires 22 cores.

Instead of parallelizing SubBytes-16 into four SubByte-4s, we can replace it with 16 SubBytes-1s. The effective execution delay of the SubBytes process is reduced to 10 cycles. As a result, the latency of one-loop decreases to 120 cycles. Therefore, the throughput of the cipher is increased to 67.5 cycles per byte. However, it requires seven additional cores dedicated to communication (four Merge Cores and three Dispatch Cores), which impair the area and energy efficiency of the implementation.

Full-Parallelism

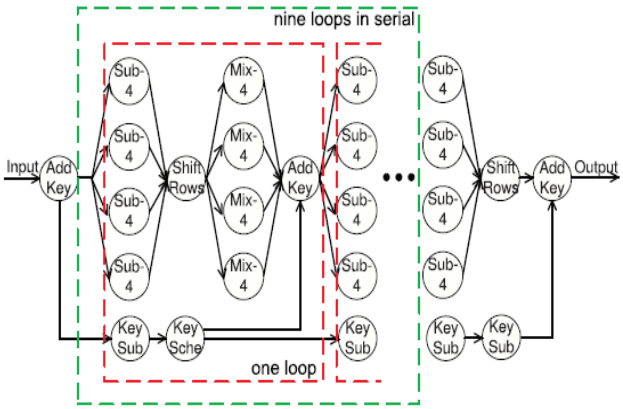


Fig. 7.1 Full Parallelism

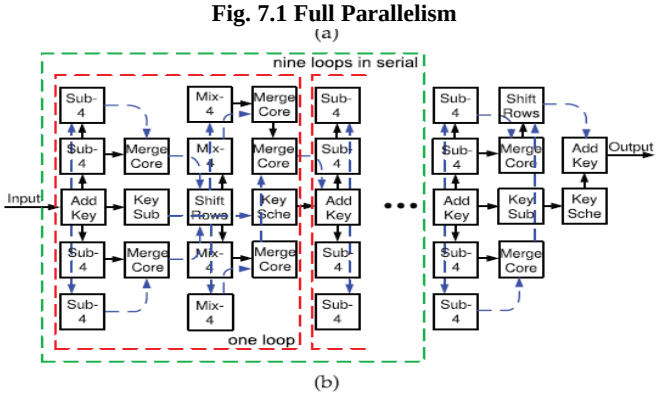


Fig. 7.2 Full Parallelism

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 3, May 2014

The Full-parallelism AES implementation combines the Parallel-SubBytes-Mix Columns model and loop unrolling. The dataflow diagram and the mapping of the Full-parallelism model are shown in Figure 7. As expected, the throughput of this design is the highest among all of the models introduced in this paper since it employs most data and task parallelism. The throughput of the Full-parallelism model is 70 cycles per block, equaling 4.375 cycles per byte. It also requires 164 cores, which is the largest implementation of all. In the Full-parallelism model, the MixColumns-4 processors are the throughput bottlenecks which determine the performance of the cipher. Therefore, parallelizing the SubBytes process with more than four processors would only increase the area and power overhead without any performance improvement.

IV. AREA EFFICIENCY ANALYSIS

Area is a significant metric in system design. Less area means less silicon, therefore less cost. From a many-core processor perspective, area is represented by the number of cores required to implement applications. Smaller area translates into fewer used cores and leaves more opportunities for dealing with other applications on the same platform simultaneously.

Area Optimization Methodology

Before comparing area efficiency among different AES implementations, area optimization is applied to all of the models without impairing performance. In this section, the area optimization methodology is illustrated through a detailed example of minimizing the number of cores used by the Full-parallelism model. There are 17 cores in one loop of the Full-parallelism mapping, including five communication-dedicated cores, which are used for routing only and the final round operation requires 11 cores.

Two optimization steps are applied to the Full-parallelism model. First, since the Shift Rows process is only byte-rotation, alternating the sequence of the SubBytes and the Shift Rows stages would not affect encryption results. How-ever, this alternation reduces two Merge Cores for each loop. As a result, 18 cores are reduced from the Full-parallelism model. Second, the throughput of the Full-parallelism model is 70 cycles per block, which is determined by the operation delay of MixColumns-4s. Any actors with less execution delay would not impair the performance of the system.

Area Efficiency Comparison

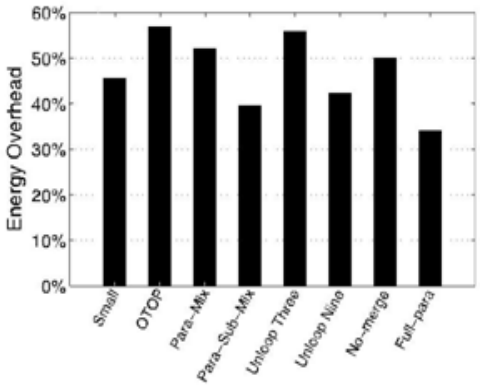
Based on the optimization methods discussed above, the number of cores utilized for each implementation is optimized as follows:

1. Small: Optimization methods are not applicable.
2. OTOP: The SubBytes and Shift Rows processors in the last round are fused into one processor, saving one processor.
3. Parallel-Mix Columns: The SubBytes and Shift Rows processors in the last round are fused into one processor, saving one processor.
4. **Parallel-SubBytes-MixColumns**: The sequence of the SubBytes and the Shift Rows stages is alternated, which saves three processors. The SubBytes and Shift Rows processors in the last round are fused into one processor, saving one more processor.
5. **No-merge-parallelism**: Optimization methods are not applicable.

V. COMPARISON

Since the AES ciphers presented in this work are implemented on a programmable platform without any application-specific hardware, we compare our work with other software implementations on programmable processors, and do not compare with implementations that contain or are composed of specialized hardware (e.g., ASICs, ASIPs, FPGAs, etc.). AES hardware implementations have been reported to achieve throughputs per area up to tens and even hundreds of Gbps=mm^2 [9] and energy efficiencies in the range of several pJ/bit—they are in an entirely different class both in efficiencies achieved and in the cost and effort required to design.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 3, May 2014



VI. CONCLUSION

Different AES cipher implementations with both online and offline key expansion on a fine-grained many-core system is presented. Each implementation exploits different levels of data and task parallelism. The smallest design requires only six processors, equaling 1:02 mm^2 in a 65 nm fine-grained many-core system. The fastest design achieves a throughput of 4.375 cycles per byte, which is 2.21 Gbps when the processors are running at a frequency of 1.2 GHz. We also optimize the area of each implementation by examining the workload of each processor, which reduces the number of cores used as much as 18 percent. The design on the fine-grained many-core system achieves energy efficiencies approximately 2.9-18.1 times higher than other software platforms, and performance per area on the order of 3.3-15.6 times higher. Overall, the fine-grained many-core system has been demonstrated to be a very promising platform for software AES implementations.

REFERENCES

[1] Bin Liu, Member, IEEE, and Bevan M. Baas, Senior Member, IEEE "Parallel AES Encryption Engines for Many-Core Processor Arrays".

[2] Gueron S., Jan.(2010)"Intel Advanced Encryption Standard Instructions Set,".

[3] Hennessy J.L. and Patterson D.A., (2007). Computer Architecture: A Quantitative Approach, fourth ed. Morgan Kaufmann.

[4] NIST, "Nov.(2001). Advanced Encryption Standard (AES),"/publications /fips197/fips -197.pdf.

AUTHOR BIOGRAPHY

V.Saminathan completed his BE Degree in the year 2003 at Government College of Technology Coimbatore and ME in the year 2008 at Maharaja Engineering College and currently pursuing PhD at Anna University in the area of Low Power VLSI Techniques. Sami_nv@rediffmail.com Mobile: 9943027347.

M.S.Arun completed his B.Tech in the year 2008 at Mount Zion College of Engg., Kadammanitta, Pathanamthitta and pursuing M.E. at Maharaja Engineering College, Coimbatoor under Anna University, Chennai.arunms1987@gmail.com. Mobile: 9495657511