The AES Encryption Circuit on a Reconfigurable Hardware

J. Sripornprasert, P. Chongstitvatana
Department of Computer Engineering
Chulalongkorn University
Phayathai road, Bangkok 10330, Bangkok, THAILAND
g49jsr@cp.eng.chula.ac.th, prabhas@chula.ac.th

***Abstract*-This work presents an AES encryption circuit implemented by a special type of machine called "Hardware Multiplexing" (HWMX). Hardware Multiplexing is a kind of reconfigurable embedded processor. By using dynamic reconfiguration concept, HWMX operates AES Algorithm correctly with efficient resources. HWMX consists of two cores 1) Register Bank and 2) Reconfigurable Core. The first part stores temporal results for next round computation. The second part, an important core of HWMX, processes as a reconfigurable system. This approach allows HWMX to pursue AES encryption application by splitting AES circuit into four segments and execute one segment per round, putting the temporal result of each segment into Register Bank. By using time-multiplexing hardware the resource consumed is less than a conventional circuit.**

I. INTRODUCTION

Presently, the embedded system plays critical role of computation in everyday life. At the same time, application is becoming more complex. Therefore, circuit area must be enlarged to provide more complex functionality. As a result, the circuit size could not match for budgetary environment. Hardware Multiplexing (HWMX) is a good solution for embedded systems with limited resources.

HWMX does have contribution to economic resources and its flexibility makes it suitable for embedded systems. Moreover, AES encryption is an instance of problem that we are interested in. This work demonstrates how to apply the concept of Hardware Multiplexing to implement an AES encryption circuit efficiently.

II. RELATED WORKS

The paper [1] describes the fundamental concept of Hardware Multiplexing which it provided the simulation result. This work also provides the simulation result. The simulation compares between HWMX and a general processor which illustrates HWMX's advantages.

In [2], a reconfigurable AES core using FPGA is represented. Their work presents a reconfigurable component for saving resources, S-Box. The S-Box is an important part of ByteSub process. It uses large area because of ByteSub look-up table (Encryption/Decryption). To reduce ByteSub look-up table area, a reconfigurable S-Box would be used ByteSub value stored in S-Box counts on whether it is Encryption or Decryption. This concept saves 50% of overall resource area. However, the reconfiguration of entire data path (not only S-Box) which relies upon current function is the core concept of HWMX.

Another work [3], proposed a design of Reconfigurable AES Encryption/Decryption for mobile terminals. It made up of Reconfigurable Crypto-Unit (RCU) and Reconfiguration Control Logic (RCL). The RCU has the multiplier which supports encryption, decryption, error correction and error detection. The RCU obtain control signals from the RCL. The RCL is reconfigured by Control Unit. Consequently, the RCU and the RCL give overall system more flexibility.

When a computation task is larger than the available area, it is needed to be partitioned. The work [4] proposed methods to partition a design into reconfigurable blocks. This work states two partition methodologies, Level-Based partition and Clustering-Based partition. If Level-Based partition is applied to a task, the independent subtasks will be put into reconfigurable area. On the other hand, The Clustering-Based partition will put dependent subtasks into reconfigurable area. As a result, Level-Based partition spends a little time of execution delay but it has expensive communication time and vice versa for Clustering-Based partition.

III. OPENCORE AES IP-CORE

We compare our work with an Opencore AES engine. The work [6] meets Rijndael's specification and has publicly available information (Fig. 1).

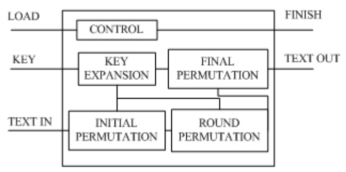

Figure 1. Opencore AES architecture which finishes its task by 12 cycles. It starts when completely load 128-bits key and 128-bits data into machine.

Opencore AES IP-Core has five modules integrated into a structure. There are four modules that are connected to each other. A group of modules does transformation and computes a RoundKey per each round. The Opencore Encryption cipher consumes chip area about 40,000 equivalent gates. This synthesized result comes from FPGA Xilinx spartan3 xcv200.

IV. Overview of AES Encryption and Hardware Multiplexing

This section describes a basic background of AES Encryption. Moreover, some fundamental concept of how to apply HWMX to the AES core would be described.

A. AES Encryption Background

AES encryption [5] is an iterated block cipher with a variable block length and a variable key length. The block length can be independently specified to 128, 192 or 256 bits. The round transformation is composed of four different transformations. From Rijndael's proposal we obtain Table I.

TABLE I
For 1st -to 9th of AES Encryption Uses Left Side Table, but Final Round Uses Right Side.

Round(State, RoundKey){
    ByteSub(State);
    ShiftRow(State);
    MixColumn(State);
    AddRoundKey(State);
}
	FinalRound(State,RoundKey){
	    ByteSub(State);
	    ShiftRow(State);
	    AddRoundKey(State);
	}

* The intermediate cipher result is called *State.*

B. Hardware Multiplexing Architecture

HWMX contains two essential components, Register Bank (RB) and Reconfigurable Data Path (RDP), as show in Fig. 2.

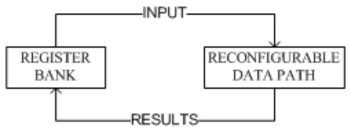

Figure 2. RB and RDP work together by sharing input, output and temporal result.

From Fig. 2, HWMX manages process cycles around RB and RDP. RB keeps three types of data, input, output and temporal results. These data comes from RDP's unit, which links with Control Unit. The Control Unit outputs fix sequence of control signal to RB and the Control Unit issues commands to the sequence ROM as a pointer to the desired sequence. (See Fig. 3)

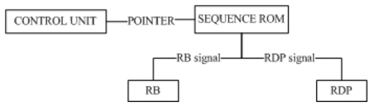

Figure 3. Control Unit send pointer to sequence ROM for RB signal and RDP signal.

The RDP realized the AES circuit by dividing it into four modules, Transposer, KeyExpansion, cipherSB and cipherMX. At any time, the RDP will be configured as one of these modules (Fig.4).

As you see in Fig. 4, there are two types of execution that is Initial Round and Others Round (1st-10th Round). Initial Round is a round which provides data ready to be executed in next round. In 1st-10th Round, all of processes will change. These rounds calculate the final result when they reach 10th round. Execution sequence is discussed in detail again in EXECUTION section.

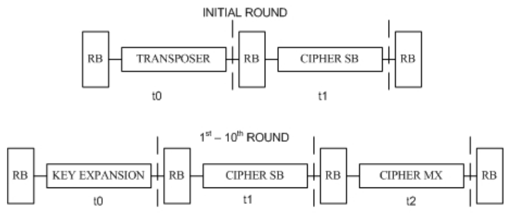

Figure 4. AES Encryption instance has two types of loop that has to be executed along the encryption.

V. Implementation

This section presents briefly the details of HWMX architecture and then shows the AES Encryption. HWMX has two main cores. There are Double Register Banks and Reconfigurable Data Path, called RDP.

Double Register Banks: Each bank contains sixteen 32-bit registers. These cores are designed to store any temporal results that come from RDP and then RDP will fetch these temporal results to compute next results.

Reconfigurable Data Path (RDP): RDP has a self-reconfigurable capability. It changes its circuit to maximize utilization and minimize resource usage. The idea to achieve this encryption by minimum resources consumption is to break an entire circuit into smaller fragments. If a fragment is too large that means circuit spends expensive resource but if a fragment is too small that means circuit wastes time. In this paper, AES Encryption circuit is divided into four modules, a Transposer, a CipherSB, a CipherMixColumn and a KeyExpansion. We assume method of reconfiguration by using Multiplexer. Reconfiguration happens when Multiplexer swaps itself to connect to other modules. By this assumption, the reconfiguration time is zero. Because there is no known practical and fast reconfiguration method at present, this assumption is amounted to a best case scenario.

VI. Experiment

We illustrate how an AES circuit based on HWMX work by comparing it to two conventional designs, one in based on a general purpose processor and the second one is an AES engine.

Traditional CPU: the CPU fetches instructions, and then decodes them. After that, The CPU executes instructions. All of these steps run on the same data path, no matter what operations are.

The AES engine: the special purpose AES engine has its circuit designed specific for the task. It has fast data-flow style execution which can be parallelized. It can be much faster than a general purpose processor trade in by the size of the circuit.

The AES based on HWMX: a HWMX circuit changes its data path (the reconfigurable unit) from time to time to share resource among different part of computation. It can trade off the speed with the size of the circuit. Table II shows the sequence of multiplexing the reconfigurable units. This is a sequence of data paths that it uses to find the result. (See Table II)

TABLE II
TABLE II SHOWS THE SEQUENCE HWMX OF EXECUTION. INITIAL ROUND PREPARES DATA INPUT READY TO PROCESS NEXT ROUND. 1st-10th ROUND DOES ENCRYPTION TO FIND OUTPUT.

Initial Round	1st-10th Round
Transposer cipherSB	KeyExpansion cipherSB cipherMX*

*cipherMX in final round would not do mix column operation.

First of all, RDP selects the Transposer module (Fig. 5) to be used. It fetches input data and input key from RB. After it finishes matrices transpose operation ,it stores the results into RB. The cipherSB module (Fig. 6) replaces the Transposer module. To prepare encryption, the cipherSB does addRoundKey. When it gets the end of addRoundKey process, RDP begins encryption. The AES encryption iterates the intermediate results for 10 rounds (see Table II). Each round has the same sequence. Firstly, the KeyExpansion module (Fig. 7) is the first module which RDP selects. The KeyExpansion module causes a new Round Key that other modules need to manipulate the Round Cipher. The KeyExpansion module finds a Round Key, and then it places them into RB. Thereafter, the cipherSB would replace the KeyExpansion module. Instead of doing addRoundKey process again, the cipherSB does ByteSub. The intermediate results from the cipherSB would be stored to RB. Finally, the cipherSB transforms itself to the cipherMX module (Fig. 8). At the beginning, the cipherMX module fetches data, and then executes shiftRow, mixColumn and addRoundKey. These set of operation which consist of the KeyExpansion, the cipherSB and the cipherMX module would be executed that sequence until they reach 10th round encryption. Furthermore, the final result is store into RB. Fig. 5-8 illustrate each component.

VII. CONCLUSION

The following paragraphs compare among HWMX, Traditional CPU and AES engine. Any advantages and disadvantages would be discussed.

TRANPOSER
INPUT
MATRIX TRANSPOSE
OUTPUT

Figure 5. Inside the Transposer module, matrix transpose module transforms input to ready-to-use form.

Traditional CPU: a general purpose processor has its performance disadvantage due to its generic data path. However, the processor can perform other general purpose task as well.

The AES engine: an AES engine is built to be specific encryption hardware. Certainly, this approach has highest performance. On the other hand, it consumes the largest area when compares to others and can not do anything but AES Encryption.

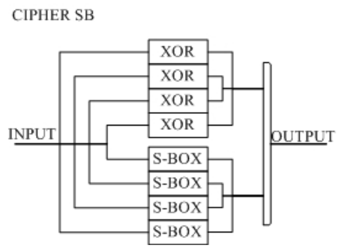

Figure 6. Inside cipherSB module, there are four S-Boxes and four XORs module to find ByteSub result.

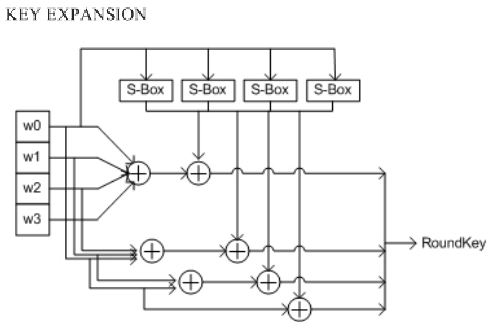

Figure 7. KeyExpansion module is the most complicated module, so this reason is why KeyExpansion is the largest part.

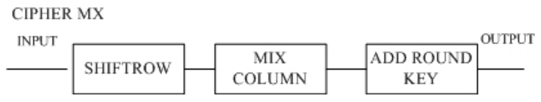

Figure 8. The cipherMX module consists of three sub-modules to do three operations, ShiftRow, MixColumn and AdddRoundKey.

The AES based on HWMX: HWMX does the AES encryption by RDP. RDP has the advantage of a flexible data path by reconfigurable ability. The ability of partition hardware into a smaller fragment helps to overcome the resource problem. Although the speed of HWMX is traded off with the resource, it can be designed to be “in-between” the performance of the general purpose processor and the special engine. This flexibility is shown in Table III.

This paper demonstrates a design of AES circuit based on HWMX concept. The comparison is made between the proposed design, AES engine, and a general purpose processor, Intel 8051 [8], which has a published performance figure. From Table III, HWMX spend around 10 thousand gates less than AES engine three times and two times for Intel 8051 [5]. The reason that HWMX is smaller than any other because its core, RDP. However, its speed is still slower than the specific AES Encryption engine. The Bottleneck

Communication is a major problem according to nature of Level-Based partition. Each module that RDP change operates on 128-bits bus but HWMX bus is Double 32-bits bus. Therefore, it wastes 2 cycles to load data into each module. Nevertheless, if HWMX has 128-bits bus instead, it will be much faster. Anyway, that is not the sole purpose of HWMX. HWMX idea is more general than a circuit that does only AES encryption.

TABLE III
RESOURCES AND CYCLES USAGE SUMMARY SHOW THAT HWMX TAKE AN ADVANTAGE ON EQUIVALENT GATES.

	Equivalent gates	128-bit Encryption Cycles	Freq (MHz).
HWMX	10830	202	75.644
AES engine	40621	12	137.299
Intel 8051	38203* 37902**	48780* 4878**	12.00* 18.906**

* original 8051 [5], ** Oregano 8051 [8]

VIII. FUTURE WORKS

In this paper, HWMX never do the physical dynamic self-reconfigurable circuit. The technology of hardware has not reaches the actual self-reconfigurable yet. Presently, we are working on a study of integrating two Xilinx spartan3 boards, one is master and other is slave. Master board consists of Double Register Banks and Microblaze. Slave board is reserved only for RDP. Master board handles storing temporal results and request to Microblaze for reconfiguration. Microblaze, then, put bitstream that store in it memory into slave board via JTAG.

REFERENCES

[1] K. Piromsopa, P Bavonparadon and P Chongstitvatana, "Hardware multiplexing: towards a resource efficient reconfigurable processor." *3rd International Symposium on Communications and Information Technologie*s, Songkhla, Thailand, 2003

[2] X. Jian, L.Yuan-Feng, D. Zi-Bin and S. Yi, "Design and Implementation of AES IP-Core Using FPGA," *ASIC, 2005. ASICON 2005. 6th International Conference On*, Shanghai, China, 2005, pp.765-768.

[3] T. Pionteck, T. Staake, T. Steifmeir, L.D. Kabulepa and M. Glesner, "Design of Reconfigurable AES Encryption/Decryption Engine for Mobile Terminals," *Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on*, Vancouver, BC, Canada, 2004, II- pp.545-8.

[4] K.M.G. Purna and D. Bhatia, "Temporal and Scheduling Data Flow Graphs for Reconfigurable Computers," *Computers, IEEE Transactions on*, 1999, pp.579-590.

[5] National Institute of Standards and Technology (NIST), FIPS Publication 197, "Advanced Encryption Standard (AES)," November 2001.

[6] R. Usselmann, AES IP-Core, www.opencore.org, 2002.

[7] K. Compton and S. Hauck, "Reconfigurable Computing: A Survey of System and Software," ACM Computing Survey, NY, USA, 2002, 171-210.

[8] http://www.oregano.at/ip/ip01.htm.