LETTER

Design and Evaluation of Data-Dependent Hardware for AES Encryption Algorithm**

Ryoichiro ATONO[†*], ***Nonmember and*** **Shuichi ICHIKAWA**[†,††a)], ***Member***

SUMMARY If a logic circuit was specialized to a specific input, the derived circuit would be faster and smaller than the original. This study presents various designs of a key-specific AES encryption circuit. In our iterative design, 41% of the logic gates and 20% of RAM were reduced, while 24% more performance was derived. In our pipelined design, 54% of the logic gates and 20% of RAM were reduced, while 74% higher performance was achieved. The results on DES encryption circuits are also presented for comparison.

key words: *FPGA, custom circuit, partial evaluation, specialization, cryptography, embedded system*

1. Introduction

Today, embedded systems are integral parts of our everyday lives. To fulfill various requirements that sometimes contradict each other, reconfigurable devices (including FPGAs) have been widely adopted in embedded systems. Reconfigurable devices are not poor substitutes for traditional ASICs, but can outperform ASICs if the system design were optimized for these devices. This study investigates a design technique which takes advantage of reconfigurability.

In embedded systems, it is essential to reduce logic scale, because the reduction of logic gates results in the reduction of cost and power consumption. The logic circuit can generally be reduced, if any input of the circuit is given as a constant. The derived circuit would be dependent on input data, and thus designated as *data-dependent hardware* in this study. A similar technique is called *partial evaluation* or *specialization* in software [1]. It is obvious that data-dependent hardware is smaller in logic scale and higher in operational frequency than the original, because both logic scale and logic depth can be reduced. An obvious drawback is that it has to be generated for each input instance. This naturally means that reconfigurable devices are best suited to implement data-dependent hardware, where partial reconfiguration is a favorable option for practical applications. It is worth noting that the circuit generation time has great practical importance.

The purpose of this study is to evaluate the effects of data-dependent circuits for AES algorithm [2]. More specifically, this study quantitatively presents the reduction of logic scale, the improvement of throughput, and the circuit generation time for AES encryption circuits of iterative and pipelined designs. The results on DES encryption circuit will also be presented for comparison. Since encryption and decryption are very similar in both AES and DES, we examine only encryption circuits in the following discussion.

2. Related Studies

Cryptographic circuits are important components for many embedded systems, and thus there are many studies on cryptographic circuit designs for FPGAs. More information on this subject can be found in a comprehensive survey by Wollinger et al. [3]. Among the preceding works, there are only a few studies on *data-dependent* cryptographic circuits.

Leonard and Mangione-Smith [4] presented the evaluation results on partially-evaluated DES circuits. They examined the key-specific versions of iterative DES circuits on Xilinx XC4000 and AT&T OR2C FPGA, and reported that the logic scale was reduced to 55% of the original, while the data rate was 31% better than the original. Though their work is encouraging, some problems remain: First, they did not show the quantitative evaluation results of circuit generation time, but only stated that this technique is applicable when the duration of a session key is longer than the circuit generation time. Second, they did not examine pipelined designs, which provide higher throughput than iterative designs. The last problem is that DES is now obsolete and has been replaced by a new standard, AES [2].

Payne [5] presented the data-dependent circuits for finite field operations (constant multiplication and fixed polynomial division over $GF(2^k)$), which are applicable to elliptic curve cryptosystems. Taylor and Goldstein [6] investigated the data-dependent designs for various ciphers, particularly for the IDEA block cipher. Patterson [7] discussed dynamic circuit specialization of DES circuits based on a specific key. He [8] also examined key-specific circuits for the Serpent block cipher.

Although the AES circuit is practically very important, data-dependent designs for the AES cipher have not been examined to date. This is the first quantitative report on data-dependent AES circuits.

Manuscript received August 30, 2005.
Manuscript revised March 27, 2006.

[†]The authors are with the Department of Knowledge-based Information Engineering, Toyohashi University of Technology, Toyohashi-shi, 441–8580 Japan.
[††]The author is with the Intelligent Sensing System Research Center, Toyohashi University of Technology, Toyohashi-shi, 441–8580 Japan.
[*]Presently, with HAL Laboratory, Inc.

[**]This work partially appeared as an extended abstract in the 2005 Annual Meeting Record IEE Japan, vol.3, pp.91–92 (March 2005).

a) E-mail: ichikawa@tutkie.tut.ac.jp

DOI: 10.1093/ietisy/e89–d.7.2301

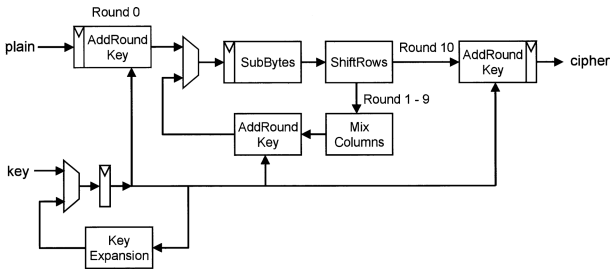

Fig. 1 Block diagram of "original" design.

3. Design

AES [2] is a block cipher adopted as an encryption standard by the U.S. government. In this study, the AES circuit by Usselmann [9] was adopted as the basis of evaluation. Usselmann's design is a portable and straight-forward implementation of the AES standard, which makes it suitable as the basis of evaluation. Usselmann's design is written in VerilogHDL, and provided as an open source IP core. The session key of this circuit is fixed to 128 bit length.

Figure 1 illustrates the block diagram of Usselmann's encryption circuit (**original**), in which a 128-bit plain text is encoded in 11 cycles. In Fig. 1, KeyExpansion generates 11 round keys from the session key—one for the initialization, nine for the iteration, and one for the finalization. A series of operations (SubBytes, ShiftRows, MixColumns, and AddRoundKey) are repeated 9 times.

If EDA tools were all-powerful, good data-dependent circuits would be generated by just giving constant values to EDA tools with the original design. However, as shown in Sect. 4, EDA tools are not that powerful. Therefore, we prepared the following iterative designs, in addition to the original design, to examine the resulting circuits. All these designs are functionally equivalent, except that the expressions in HDL are different from each other.

original Usselmann's encryption circuit (Fig. 1). This is *not* a data-dependent circuit; i.e., "key" is variable.

fixed_key The circuit derived by giving a constant "key" to the "original" design. Optimization of logic is wholly left to the EDA system.

fixed_round_key If the session key is known to be constant, 11 round keys can be calculated beforehand. Thus, KeyExpansion in Fig. 1 can be replaced by a multiplexer of 11 constants.

xor_collapse AddRoundKey in Fig. 1 performs bitwise *xor* operations of input data and round key. If "key" is constant, these *xor* operations can be reduced to wires or inverters. The multiplexer in KeyExpansion is also integrated with AddRoundKey.

xor_by_ROM_all If "key" is constant, AddRoundKey could be a simple combinatorial circuit whose inputs are text data and round number. Thus, AddRoundKey units for Round 0–10 can be replaced by a ROM.

xor_by_ROM_partial As readily seen from Fig. 1, Rounds 0 and 10 have different dataflows from Rounds 1–9. Thus, in this design, AddRoundKey units for Rounds 0 and 10 are implemented as in xor_collapse, while AddRoundKey for Rounds 1–9 is implemented with a ROM.

Though the abovementioned circuits encrypt 128-bit text in 11 cycles, they can be pipelined for improved performance. We also designed pipelined circuits of 11 stages, which correspond to the iterative designs described above. These pipelined designs are designated by the prefix "**pl_**" in the following discussion. The pipelined versions of "xor_by_ROM_all" and "xor_by_ROM_partial" were not evaluated, because (1) they require too many ROMs to fit in our target device, and (2) the throughputs of xor_by_ROM_all and xor_by_ROM_partial are inferior to those of fixed_round_key and xor_collapse as shown in Sect. 4.

It might be possible to develop a dedicated software to design and implement data-dependent AES circuits. Such dedicated software could be used with existing EDA tools to improve circuit quality and generation time. However, in this study, we adopted standard EDA tools solely to concentrate on a baseline evaluation, while leaving dedicated software for future studies.

4. Evaluation

The designs described in the previous section were implemented and evaluated on Xilinx Virtex-II architecture [10]. The target device was set to XC2V4000-FF1252-4, which contains 23040 Slice and 120 BlockRAM. We used Leonardo Spectrum (ver. 2005a.82) for logic synthesis, and Xilinx ISE 8.1.01i for mapping, placement, and routing. Optimization level was left to default value (auto). All EDA tools were executed on an AMD Athlon 64 3500+ processor with 2 GB memory (Windows 2000 Professional SP4).

Table 1 summarizes the evaluation results of data-dependent AES circuits. Obviously, the value of the session key affects the size and performance of the consequent data-dependent circuit. Therefore, the results shown in Table 1 are the average values of the circuits of 100 random keys, except the design "original". The design "original" is not data-dependent, and thus its results are uniquely determined.

Usselmann's design is portable, but not optimized for Xilinx devices. Therefore, we slightly modified the original source code, and implemented SubBytes using the BlockRAM of Virtex-II FPGA [10]. This greatly reduces the logic scale in exchange for 10 BlockRAMs, as shown in Table 1. All other designs were derived from this modified "original" circuits with BlockRAM.

The design "fixed_key" is slightly smaller and slower than the "original", but the differences are not large. This fact proves that EDA tools are not all-powerful, which justifies the following design variations. The design "fixed_round_key" is 17% smaller in slices, 20% smaller

Table 1 Circuit generation results of AES circuits.

Design	Gen. Time (sec.)	Logic Scale (slice)	Block RAM	Max. Freq. (MHz)	Throughput (Mbps)
original (w/o BlockRAM)	204.	1777	0	83.4	970.
original	108.	449	10	73.7	858.
fixed_key	104.	442	10	71.4	831.
fixed_round_key	100.	371	8	90.8	1057.
xor_collapse	106.	265	8	91.3	1062.
xor_by_ROM_all	104.	255	27	40.6	472.
xor_by_ROM_partial	95.	192	27	45.9	534.
pl_original	6455.	2570	100	51.8	6630.
pl_fixed_key	6414.	2424	100	61.4	7859.
pl_fixed_round_key	1687.	1192	80	90.2	11546.
pl_xor_collapse	1685.	1192	80	90.2	11546.

Table 2 Estimated operational frequencies of AES circuits with/without wiring delay.

Design	Wiring delay inclusive (MHz)	No wiring delay (MHz)	Ratio
original (w/o BlockRAM)	83.4	192.8	0.433
original	73.7	105.6	0.698
fixed_key	71.4	105.6	0.676
fixed_round_key	90.8	193.8	0.469
xor_collapse	91.3	183.5	0.498
xor_by_ROM_all	40.6	96.9	0.419
xor_by_ROM_partial	45.9	119.5	0.384

Table 3 Detailed circuit generation times of AES circuits (sec.).

Design	Total	Synthesis	NGDbuild	Map	Par	Trce
original (w/o BlockRAM)	204.	106.	11.	9.	72.	6.
original	108.	51.	7.	5.	41.	4.
fixed_key	104.	51.	7.	4.	38.	4.
fixed_round_key	100.	49.	7.	4.	37.	4.
xor_collapse	106.	58.	6.	4.	34.	3.
xor_by_ROM_all	104.	55.	8.	4.	34.	3.
xor_by_ROM_partial	95.	48.	8.	4.	32.	3.
pl_original	6455.	6313.	14.	12.	108.	8.
pl_fixed_key	6414.	6273.	14.	11.	108.	8.
pl_fixed_round_key	1687.	1598.	10.	7.	66.	6.
pl_xor_collapse	1685.	1595.	10.	7.	66.	6.

in BlockRAMs, and 23% better in throughput. The design "xor_collapse" is even better—41% smaller in slices, 20% smaller in BlockRAMs, and 24% better in throughput.

The design "xor_by_ROM_all" was designed to reduce slices by implementing AddRoundKey with BlockRAM; the result was 43% smaller in slices with 2.7 times more BlockRAMs and 45% less throughput. The design "xor_by_ROM_partial" was better than xor_by_ROM_all—57% smaller in slices with 2.7 times more BlockRAMs and 38% less in throughput. Such losses of throughput are caused by the low operational frequencies of these designs, as seen in Table 1. Since the logic scales of xor_by_ROM_all and xor_by_ROM_partial are less than fixed_round_key and xor_collapse, low operational frequencies are regarded as caused by the BlockRAMs in the critical path.

Table 2 summarizes the estimated operational frequencies with and without wiring delay, which are derived from the respective design files after and before PAR (placement and routing). In Table 2, a smaller ratio suggests that the effect of wiring delay is larger. It is readily seen that the reduced operational frequencies of xor_by_ROM_all and xor_by_ROM_partial are mainly caused by additional wiring delay for the additional BlockRAMs.

Though the throughput of xor_by_ROM_all is inferior to other designs, xor_by_ROM_all is potentially interesting. Since its key-dependent part is concentrated in SRAM modules, it might be possible to implement any key-specific circuits by loading data to SRAM, without performing logic synthesis, placement, and routing. The key-specific content of SRAM could be generated very rapidly with a simple software, and thus the circuit generation time might be significantly shortened. Although this aspect is very important for practical applications, the authors leave this topic for future studies, and concentrate on the fundamental properties of data-dependent designs in this study.

The circuit generation times of iterative designs were around 100 seconds, which was about 50% of the "original without BlockRAM". This difference is naturally accountable, because "original w/o BlockRAM" requires many more slices than other designs. The details of circuit gen-

Table 4 Circuit generation results of DES circuit.

Design	Gen. Time (sec.)	Logic Scale (slice)	Block RAM	Max. Freq. (MHz)	Throughput (Mbps)
original (w/o BlockRAM)	126.	589	0	165.0	660.
original	116.	517	4	96.2	385.
fixed_round_key	71.	81	4	89.2	357.
xor_collapse	75.	102	4	73.8	295.
pl_original	177.	1356	64	68.1	4358.
pl_fixed_round_key	117.	828	64	69.9	4474.
pl_xor_collapse	116.	835	64	69.2	4429.

Table 5 Detailed circuit generation times of DES circuit (sec.).

Design	Total	Synthesis	NGDbuild	Map	Par	Trce
original (w/o BlockRAM)	126.	69.	5.	4.	44.	3.
original	116.	63.	6.	4.	39.	3.
fixed_round_key	71.	34.	6.	3.	25.	3.
xor_collapse	75.	38.	6.	3.	25.	3.
pl_original	177.	89.	9.	6.	67.	6.
pl_fixed_round_key	117.	59.	7.	5.	42.	4.
pl_xor_collapse	116.	58.	7.	5.	41.	4.

Table 6 Circuit generation results of DES circuit, where the optimization option was set to "area".

Design	Gen. Time (sec.)	Logic Scale (slice)	Block RAM	Max. Freq. (MHz)	Throughput (Mbps)
original (w/o BlockRAM)	87.	396	0	133.0	532.
original	81.	332	4	81.7	327.
fixed_round_key	68.	81	4	88.9	356.
xor_collapse	69.	103	4	74.8	299.
pl_original	158.	1357	64	67.4	4314.
pl_fixed_round_key	104.	828	64	69.8	4467.
pl_xor_collapse	105.	835	64	69.2	4429.

eration times are summarized in Table 3.

The pipelined designs "pl_fixed_round_key" and "pl_xor_collapse" were 54% smaller in slices, 20% smaller in BlockRAMs, and 74% higher in throughput. However, the circuit generation time is more than 28 minutes, which will limit the applications of these designs. Most of the generation time is consumed for logic synthesis, as readily seen in Table 3.

In both iterative and pipelined designs, data-dependent AES encryption circuits were proven to be smaller in logic scale and higher in performance, using "fixed_round_key" or "xor_collapse" design schemes.

5. The Case of DES Circuit

Our designs are not specific to only the AES circuit, but also applicable to the DES circuit. To examine our designs for the DES algorithm, we applied two of our successful schemes to the DES encryption circuit and discussed the results considering a preceding study [4]. Another aim of this section is to show the circuit generation time of data-dependent DES circuits, which was not shown by Leonard and Mangione-Smith [4].

As a basis of evaluation, we adopted Usselmann's DES circuit [11], which is an open source IP core written in VerilogHDL. This circuit is convenient for comparison, because it can be processed by the same EDA environment as his AES circuit [9].

Table 4 summarizes the evaluation results of data-dependent DES circuits. Here, again, the results of data-dependent circuits are the average values of the 100 random keys. The design "original" in Table 4 designates Usselmann's DES design.

The slices were much reduced in data-dependent iterative DES designs—84% smaller in "fixed_round_key" and 80% smaller in "xor_collapse" compared to the "original" design. This is far better than the previous results [4], where 45% was reduced. The circuit generation times were as short as 71–75 seconds (Table 5). The throughput of these designs are slightly less than the "original"—7.3% less in "fixed_round_key" and 23% less in "xor_collapse". These results do not match the 31% improvement in a previous study [4].

Though the reasons for the above differences are not elucidated, they are supposed to be caused by the following factors. Key-specific design reduces the logic of sub-key generation part. The operational frequency will be improved, if the critical path includes sub-key generation part (as in a previous study [4]). Otherwise, the operational frequency would not be improved. The second factor is the difference in optimization strategy of EDA tools. Table 6 lists the design summaries of DES circuits, where the optimization option was set to *area*, instead of *auto* of Table 4. Two

designs fixed_round_key and xor_collapse were not affected by *area* optimization, while the "original" design was much reduced in both logic scale and operational frequency. With *area* option, the differences in throughput are much reduced, where the reduction of logic scale is also reduced according to 36% reduction of slices in the "original".

In pipelined DES circuits, approximately 40% of slices were reduced in Table 4, while the throughput was sustained. It is worth noting that the circuit generation time was less than 3 minutes in pipelined data-dependent DES circuits. This is almost the same as iterative AES circuits, and less than 7% of pipelined AES data-dependent circuits.

6. Conclusion

This study presented various designs of key-specific AES and DES encryption circuits. In both iterative and pipelined designs, the reduction of logic scale and the improvement of throughput were achieved using our fixed_round_key and xor_collapse designs.

In the present study, the authors concentrated on finding good designs to reduce logic scale and to boost throughput, while depending on existing EDA tools to generate key-specific circuits. This resulted in a relatively long generation time, which is about 100 seconds for iterative AES designs and 30 minutes for pipelined AES designs. Such generation time would only be acceptable for a relatively long-term use of the same key. Although the further reduction of circuit generation time is required for practical applications, it is beyond the scope of this preliminary study and must be left to future studies.

Acknowledgments

The authors are grateful to the reviewer for his helpful comments. We are also grateful to the members of Ichikawa Lab. for their various support for this work. This study was partially supported by a Grant-in-Aid for Scientific Research from the Japan Society for the Promotion of Science (JSPS). Support for this work was also provided by the 21st Century COE Program "Intelligent Human Sensing" from the Ministry of Education, Culture, Sports, Science and Technology.

References

[1] C. Consel and O. Danvy, "Tutorial notes on partial evaluation," Proc. 20th ACM Symp. on Principles of Programming Language, pp.493–501, ACM, 1993.
[2] Federal Information Processing Standards Publication 197, "Specification for the advanced encryption standard (AES)." http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
[3] T. Wollinger, J. Guajardo, and C. Paar, "Security on FPGAs: State-of-the-art implementations and attacks," ACM Trans. on Embedded Computing Systems, vol.3, no.3, pp.534–574, Aug. 2004.
[4] J. Leonard and W.H. Mangione-Smith, "A case study of partially evaluated hardware circuits: Key-specific DES," Proc. Field-Programmable Logic and Applications (FPL'97), LNCS 1304, pp.151–160, Springer, 1997.
[5] R. Payne, "Run-time parameterised circuits for the Xilinx XC6200," Proc. Field-Programmable Logic and Applications (FPL'97), LNCS 1304, pp.161–172, Springer, 1997.
[6] R.R. Taylor and S.C. Goldstein, "A high-performance flexible architecture for cryptography," Proc. Cryptographic Hardware and Embedded Systems (CHES'99), LNCS 1717, pp.231–245, Springer, 1999.
[7] C. Patterson, "High performance DES encryption in Virtex FPGAs using JBits," Proc. Field-Programmable Custom Computing Machines (FCCM2000), pp.113–121, IEEE Computer Society, 2000.
[8] C. Patterson, "A dynamic FPGA implementation of the Serpent block cipher," Proc. Cryptographic Hardware and Embedded Systems (CHES2000), LNCS 1965, pp.141–156, Springer, 2000.
[9] R. Usselmann, "AES (Rijndael) IP core: Overview." http://www.opencores.org/projects.cgi/web/aes_core/
[10] Xilinx Inc., Virtex™-II Platform FPGAs: Introduction and Overview, 2002.
[11] R. Usselmann, "DES/Triple DES IP cores: Overview." http://www.opencores.org/projects.cgi/web/des/