Fast implementation and fair comparison of the final candidates for Advanced Encryption Standard using Field Programmable Gate Arrays

Kris Gaj and Pawel Chodowiec

George Mason University, Electrical and Computer Engineering,
4400 University Drive, Fairfax, VA 22030, U.S.A.
kgaj@gmu.edu, pchodowi@gmu.edu

Abstract. The results of fast implementations of all five AES final candidates using Virtex Xilinx Field Programmable Gate Arrays are presented and analyzed. Performance of several alternative hardware architectures is discussed and compared. One architecture optimum from the point of view of the throughput to area ratio is selected for each of the two major types of block cipher modes. For feedback cipher modes, all AES candidates have been implemented using the basic iterative architecture, and achieved speeds ranging from 61 Mbit/s for Mars to 431 Mbit/s for Serpent. For non-feedback cipher modes, four AES candidates have been implemented using a high-throughput architecture with pipelining inside and outside of cipher rounds, and achieved speeds ranging from 12.2 Gbit/s for Rijndael to 16.8 Gbit/s for Serpent. A new methodology for a fair comparison of the hardware performance of secret-key block ciphers has been developed and contrasted with methodology used by the NSA team.

1. Introduction

Advanced Encryption Standard (AES) is likely to become a de-facto worldwide encryption standard commonly used to protect all means of secret communications during the next several decades [1]. Ever growing speed of communication networks, combined with the high-volume traffic and the need for physical security, creates a large demand for efficient implementations of AES in hardware.

The efficiency of hardware implementations of the AES candidates has been one of the major criteria used by NIST to select the new federal standard from among five final candidates. In the absence of any major breakthroughs in the cryptanalysis of final candidates, and because of the relatively inconclusive results of their software performance evaluations, hardware evaluations presented during the Third AES conference [2] provided almost the only quantitative measure that clearly differentiated AES candidates. The importance of this measure was reflected by a survey performed among the participants of the AES conference, in which the ranking of the candidate algorithms [2] coincided almost exactly with their relative speed in hardware (compare Fig. 1 with Figs. 9 and 11). In October 2000, NIST announced its
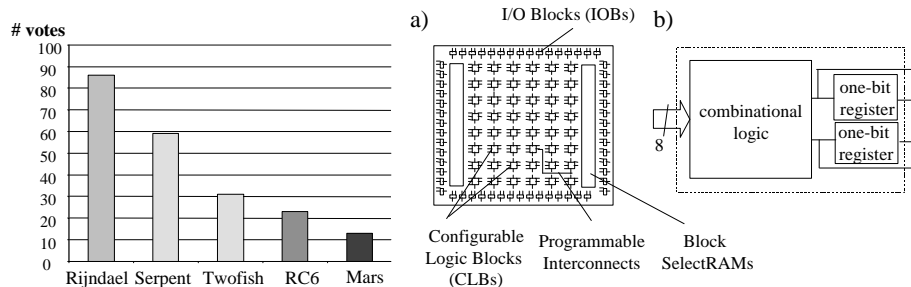
Fig. 1. Ranking of the AES candidates according to the survey performed among participants of AES3 conference.

Fig. 2. FPGA device: a) general structure and main components, b) internal structure of a Configurable Logic Block slice.

selection of Rijndael as the winner of the AES contest. The NIST final report confirmed the importance of the hardware efficiency studies [3].

The issue of implementing AES candidates in hardware will remain important long after the AES selection process is over. The winner of the AES contest, Rijndael, will be in common use for many years. The remaining AES finalists are likely to be included in products of selected vendors. New architectures developed as a part of the AES candidate comparison effort will be used in implementations of other secret-key block ciphers.

In this paper, we focus on implementing and comparing AES candidates using the reconfigurable hardware technology based on Field Programmable Gate Arrays (FPGAs). Our work supplements and extends other research efforts based on the same technology [4], [5], [6], and on the use of semi-custom Application Specific Integrated Circuits (ASICs) [7], [8], [9].

2. Field Programmable Gate Arrays

Field Programmable Gate Array (FPGA) is an integrated circuit that can be bought off the shelf and reconfigured by designers themselves. With each reconfiguration, which takes only a fraction of a second, an integrated circuit can perform a completely different function. From several FPGA families available on the market, we have chosen the high performance Virtex family from Xilinx, Inc. [10]. FPGA devices from this family consist of thousands of universal building blocks, known as *Configurable Logic Blocks* (*CLBs*), connected using programmable interconnects, as shown in Fig. 2a. Reconfiguration is able to change a function of each CLB and connections among them, leading to a functionally new digital circuit. A simplified internal structure of a *CLB slice* (1/2 of a CLB) in the Virtex family is shown in Fig. 2b. Each CLB slice contains a small block of combinational logic, implemented using programmable look-up tables, and two one-bit registers [10]. Additionally, Virtex FPGAs contain dedicated memory blocks called *Block Select RAMs*.
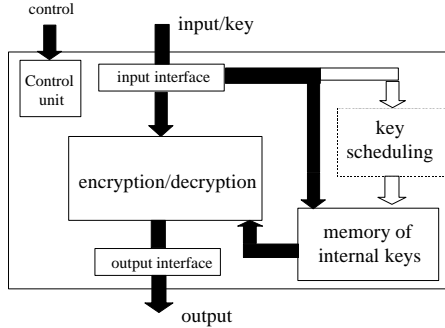
Fig. 3. General block diagram of the hardware implementation of a symmetric-key block cipher.
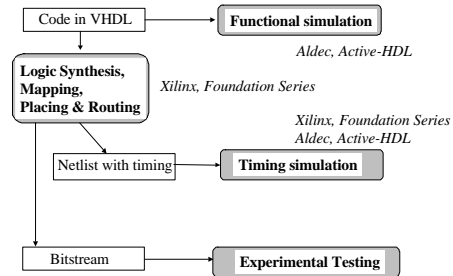

Fig. 4. Design flow for implementing AES candidates using Xilinx FPGA devices.

For implementing cryptography in hardware, FPGAs provide the only major alternative to *custom and semi-custom Application Specific Integrated Circuits* (ASICs), integrated circuits that must be designed all the way from the behavioral description to the physical layout, and sent for an expensive and time-consuming fabrication.

3. Assumptions, compared parameters, and design procedure

The general block diagram of the hardware implementation of a symmetric-key block cipher is shown in Fig. 3. All five AES candidates investigated in this paper have been implemented using this block diagram.

Our implementations are intended to support only one key size, 128 bits. To simplify comparison, the key scheduling is assumed to be performed off-chip. In order to minimize circuit area, the encryption and decryption parts share as many resources as possible by the given cipher type. At the same time, an effort was made to maximally decrease the effect of resource sharing on the speed of encryption and decryption.

The implementations of AES candidates are compared using the following three major parameters:

a. *Encryption (decryption) throughput*, defined as the number of bits encrypted (decrypted) in a unit of time.

b. *Encryption (decryption) latency*, defined as the time necessary to encrypt (decrypt) a single block of plaintext (ciphertext).

c. Circuit size (area).

The encryption (decryption) latency and throughput are related by

$$Throughput = block\_size \cdot \#\_of\_blocks\_processed\_simultaneously \, / \, Latency \qquad (1)$$

In *FPGA implementations*, the only circuit size measures reported by the CAD tools are the number of basic configurable logic blocks and the number of equivalent logic gates. It is commonly believed that out of these two measures, the number of basic configurable logic blocks approximates the circuit area more accurately.

The design flow and tools used in our group for implementing algorithms in FPGA devices are shown in Fig. 4. All five AES ciphers were first described in VHDL, and their description verified using the Active-HDL functional simulator from Aldec, Inc. Test vectors and intermediate results from the reference software implementations were used for debugging and verification of the VHDL source codes. The revised VHDL code became an input to the Xilinx toolset, Foundation Series 2.1i, performing the automated logic synthesis, mapping, placing, and routing. These tools generated reports describing the area and speed of implementations, a netlist used for timing simulations, and a bitstream to be used to program actual FPGA devices. The speed reports were verified using timing simulation.

4. Cipher modes

Symmetric-key block ciphers are used in several operating modes. From the point of view of hardware implementations, these modes can be divided into two major categories:
a. *Non-feedback modes*, such as Electronic Code Book mode (ECB) and counter mode (CTR).
b. *Feedback modes*, such as Cipher Block Chaining mode (CBC), Cipher Feedback Mode (CFB), and Output Feedback Mode (OFB).

In the non-feedback modes, encryption of each subsequent block of data can be performed independently from processing other blocks. In particular, all blocks can be encrypted in parallel. In the feedback modes, it is not possible to start encrypting the next block of data until encryption of the previous block is completed. As a result, all blocks must be encrypted sequentially, with no capability for parallel processing. The limitation imposed by the feedback modes does not concern decryption, which can be performed on several blocks of ciphertext in parallel for both feedback and non-feedback operating modes.

According to current security standards, the encryption of data is performed primarily using feedback modes, such as CBC and CFB. As a result, using current standards does not permit to fully utilize the performance advantage of the hardware implementations of secret key ciphers, based on parallel processing of multiple blocks of data [12]. The situation can be remedied by including in the NIST new standard on the AES modes of operation a counter mode and other non-feedback modes of operation currently under investigation by the cryptographic community [12].

5. Implementation of the AES candidates in feedback cipher modes

5.1 Choice of an architecture

5.1.1 Basic iterative architecture
The basic hardware architecture used to implement an encryption/decryption unit of a typical secret-key cipher is shown in Fig. 5a. One round of the cipher is implemented as a combinational logic, and supplemented with a single register and a multiplexer.

In the first clock cycle, input block of data is fed to the circuit through the multiplexer, and stored in the register. In each subsequent clock cycle, one round of the cipher is evaluated, the result is fed back to the circuit through the multiplexer, and stored in the register. The two characteristic features of this architecture are:

- Only one block of data is encrypted at a time.
- The number of clock cycles necessary to encrypt a single block of data is equal to the number of cipher rounds, *#rounds*.

The throughput and latency of the basic iterative architecture, $Throughput_{bi}$ and $Latency_{bi}$, are given by

$$Throughput_{bi} = block\_size \: / \: \#rounds \cdot clock\_period \qquad (2)$$

$$Latency_{bi} = \#rounds \cdot clock\_period \qquad (3)$$

5.1.2 Partial and full loop unrolling

An architecture with *partial loop unrolling* is shown in Fig. 5b. The only difference compared to the basic iterative architecture is that the combinational part of the circuit implements *K* rounds of the cipher, instead of a single round. *K* must be a divisor of the total number of rounds, *#rounds*.

The number of clock cycles necessary to encrypt a single block of data decreases by a factor of *K*. At the same time the minimum clock period increases by a factor slightly smaller than *K*, leading to an overall relatively small increase in the encryption throughput, and decrease in the encryption latency, as shown in Fig. 6. Because the combinational part of the circuit constitutes the majority of the circuit area, the total area of the encryption/decryption unit increases almost proportionally to the number of unrolled rounds, *K*. Additionally, the number of internal keys used in a single clock cycle increases by a factor of *K*, which in FPGA implementations typically implies the almost proportional growth in the number of CLBs used to store internal keys.

Architecture with full loop unrolling is shown in Fig. 5c. The input multiplexer and the feedback loop are no longer necessary, leading to a small increase in the cipher speed and decrease in the circuit area compared to the partial loop unrolling with the same number of rounds unrolled.

In summary, loop unrolling enables increasing the circuit speed in both feedback and non-feedback operating modes. Nevertheless this increase is relatively small, and incurs a large area penalty. As a result, choosing this architecture can be justified only for feedback cipher modes, where none other architecture offers speed greater than the basic iterative architecture, and only for implementations where large increase in the circuit area can be tolerated.

5.1.3 Resource sharing

For majority of ciphers, it is possible to significantly decrease the circuit area by time sharing of certain resources (e.g., function *h* in Twofish, 4x4 S-boxes in Serpent). This is accomplished by using the same functional unit to process two (or more) parts of the data block in different clock cycles, as shown in Fig. 7. In Fig. 7a, two parts of the data block, D0 and D1, are processed in parallel, using two

independent functional units F. In Fig. 7b, a single unit F is used to process two parts of the data block sequentially, during two subsequent clock cycles.
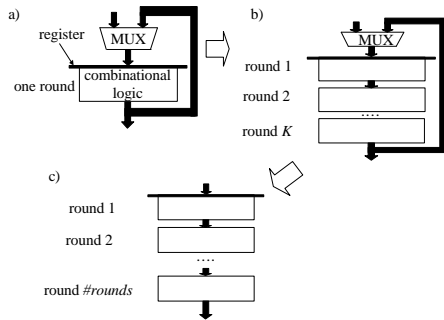

Fig. 5. Three architectures suitable for feedback cipher modes: a) basic iterative architecture, b) partial loop unrolling, c) full loop unrolling.
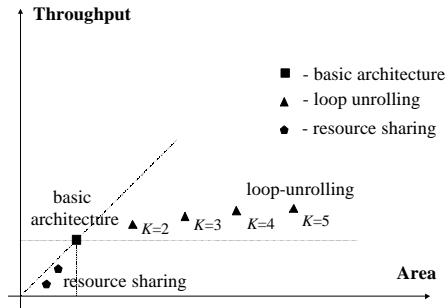
Fig. 6. Throughput vs. area characteristics of alternative architectures suitable for feedback cipher modes.
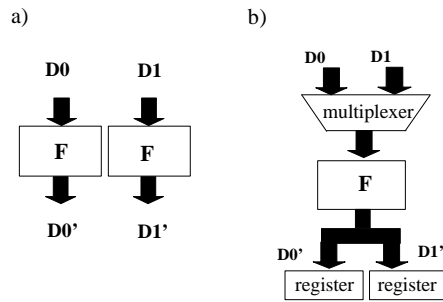

Fig. 7. Basic idea of resource sharing. a) parallel execution of two functional units F, no resource sharing, b) resource sharing of the functional unit F.
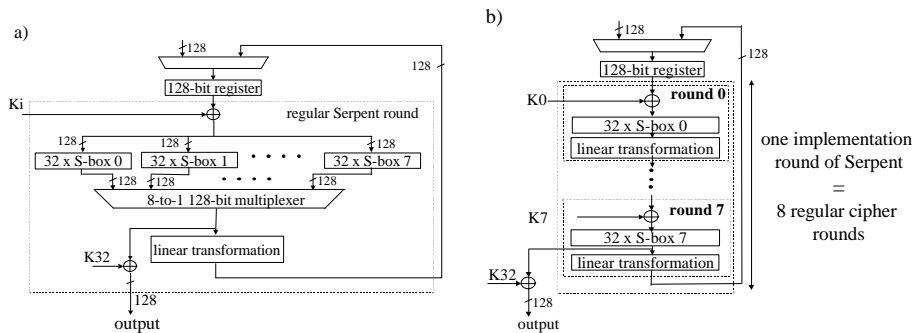

Fig. 8. Two alternative basic iterative architectures of Serpent: a) Serpent I1, b) Serpent I8.

5.1.4 Deviations from the basic iterative architecture

Three final AES candidates, Twofish, RC6, and Rijndael, can be implemented using exactly the basic iterative architecture shown in Fig. 5a. This is possible because all rounds of these ciphers perform exactly the same operation. For the remaining two ciphers, Serpent and Mars, this condition is not fulfilled, and as a result, the basic iterative architecture can be defined in several different ways.

Serpent consists of 8 different kinds of rounds. Each round consists of three elementary operations. Two of these operations, key mixing and linear transformation are identical for all rounds; the third operation, S-Boxes, is different for each of the eight subsequent rounds.

Two possible ways of defining the basic iterative architecture of Serpent are shown in Fig. 8. In the first architecture, we call Serpent I1, shown in Fig. 8a, the combinational part of the circuit performs a single regular cipher round. To enable switching between 8 different types of rounds, the combinational part includes 8 sets of S-boxes, each fed by the output from the key mixing. Based on the current round number, the output of only one of the eight S-boxes is selected using the multiplexer to feed the input of the linear transformation. In this architecture, Serpent is treated literally as a cipher with 32 rounds.

In the second architecture, we call Serpent I8, shown in Fig. 8b, eight regular cipher rounds are treated as a single *implementation round*, and implemented one after the other using a combinational logic. The implementation round needs to be computed only 4 times, to implement all 32 regular cipher rounds. Thus, in this architecture, Serpent is treated as a cipher with 4 extended cipher rounds.

Both conventions have their advantages and disadvantages. The first architecture takes less area (especially taking into account the area required for key scheduling and/or key storage). The second architecture is significantly faster.

5.1.5 Our choice

We chose to use the basic iterative architecture in our implementations. The reasons for this choice were as follows:

- As shown in Fig. 6, the basic iterative architecture assures the maximum *speed/area* ratio for feedback operating modes (CBC, CFB), now commonly used for bulk data encryption. It also guarantees near optimum speed, and near optimum area for these operating modes. Therefore it is very likely to be commonly used in majority of practical implementations of the AES candidates.
- The basic architecture is relatively easy to implement in a similar way for all AES candidates, which supports fair comparison.
- Based on the performance measures for basic architecture, it is possible to derive analytically *approximate* formulas for parameters of more complex architectures.

For Serpent, we chose to implement its basic iterative architecture shown in Fig. 8b, we refer to as Serpent I8.

5.2 Our results and comparison with other groups

The results of implementing AES candidates, according to the assumptions and design procedure summarized in section 3, are shown in Figs. 9 and 10. All implementations were based on Virtex XCV-1000BG560-6, one of the largest currently available Xilinx Virtex devices. For comparison, the results of implementing the current NIST standard, Triple DES, are also provided. Implementations of all ciphers took from 9% (for Twofish) to 37% (for Serpent I8) of the total number of 12,288 CLB slices available in the Virtex device used in our designs. It means that less expensive Virtex devices could be used for all implementations. Additionally, the key scheduling unit could be easily implemented within the same device as the encryption/decryption unit.

In Figs. 11 and 12, we compare our results with the results of research groups from Worcester Polytechnic Institute and University of Southern California, described in [4] and [5]. Both groups used identical FPGA devices, the same design tools and similar design procedure. The order of the AES algorithms in terms of the encryption and decryption throughput is identical in reports of all research groups. Serpent in architecture I8 (see Fig. 8b) and Rijndael are over twice as fast as remaining candidates. Twofish and RC6 offer medium throughput. Mars is consistently the slowest of all candidates. Interestingly, all candidates, including Mars are faster than Triple DES. Serpent I8 (see Fig. 8b) is significantly faster than Serpent I1 (Fig. 8a), and this architecture should clearly be used in cipher feedback modes whenever the speed is a primary concern, and the area limit is not exceeded.

The agreement among circuit areas obtained by different research groups is not as good as for the circuit throughputs, as shown in Fig. 12. These differences can be explained based on the fact that the speed was a primary optimization criteria for all involved groups, and the area was treated only as a secondary parameter. Additional differences resulted from different assumptions regarding sharing resources between encryption and decryption, key storage, and using dedicated memory blocks. Despite these different assumptions, the analysis of results presented in Fig. 12 leads to relatively consistent conclusions. All ciphers can be divided into three major groups:
1) Twofish and RC6 require the smallest amount of area; 2) Rijndael and Mars require medium amount of area (at least 50% more than Twofish and RC6); 3) Serpent I8 requires the largest amount of area (at least 60% more than Rijndael and Mars). Serpent I1 belongs to the first group according to [5], and to the second group according to [4].

The overall features of all AES candidates can be best presented using a two-dimensional diagram showing the relationship between the encryption/decryption throughput and the circuit area. In Fig. 13, we collect our results for the Xilinx Virtex FPGA implementations, and in Fig. 14 we show for comparison the results obtained by the NSA group for ASIC implementations [7], [8]. Comparing diagrams shown in Fig. 13 and Fig. 14 reveals that the speed/area characteristics of the AES candidates is almost identical for the FPGA and ASIC implementations. The primary difference between the two diagrams comes from the absence of the ASIC implementation of Serpent I8 in the NSA report [8].

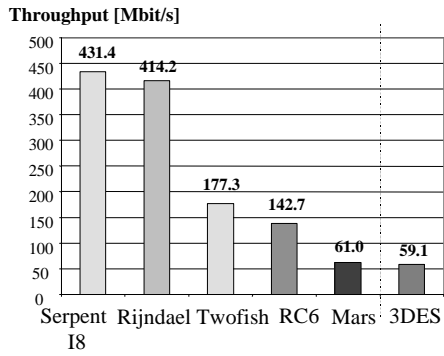All ciphers can be divided into three distinct groups:

Throughput [Mbit/s]

Fig. 9. Throughput for Virtex XCV-1000, our results.

Area [CLB slices]
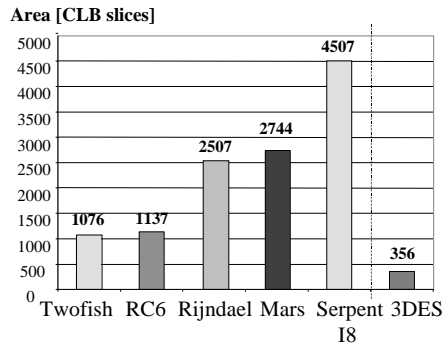
Fig. 10. Area for Virtex XCV-1000, our results.

Throughput [Mbit/s]
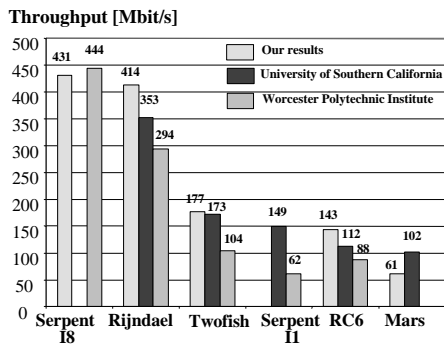
Fig. 11. Throughput for Virtex XCV-1000, comparison with results of other groups.

Area [CLB slices]
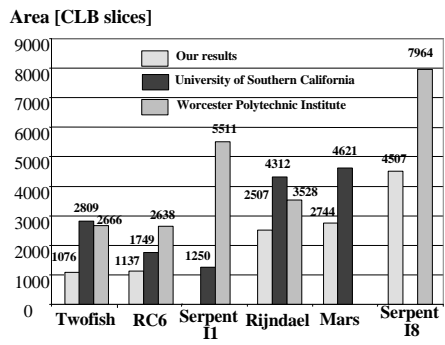
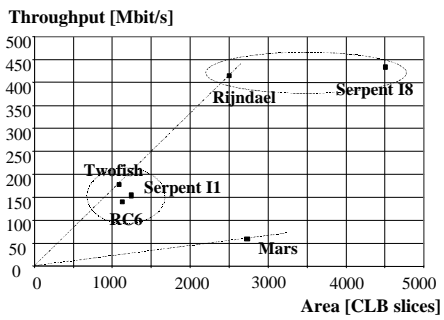**Fig. 12.** Area for Virtex XCV-1000, comparison with results of other groups.

Throughput [Mbit/s]

Fig. 13. Throughput vs. area for Virtex XCV-1000, our results. The results for Serpent I based on [5].
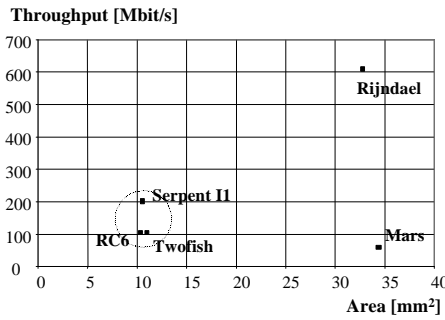
Throughput [Mbit/s]

Fig. 14. Throughput vs. area for 0.5 μm CMOS standard-cell ASICs, NSA results.

- Rijndael and Serpent I8 offer the highest speed at the expense of the relatively large area;
- Twofish, RC6, and Serpent I1 offer medium speed combined with a very small area;
- Mars is the slowest of all AES candidates and second to last in terms of the circuit area.

Looking at this diagram, one may ask which of the two parameters: speed or area should be weighted more during the comparison? The definitive answer is *speed*. The primary reason for this choice is that in feedback cipher modes it is not possible to substantially increase encryption throughput even at the cost of a very substantial increase in the circuit area (see Fig. 6). On the other hand, by using resource sharing described in section 5.1.3, the designer can substantially decrease circuit area at the cost of a proportional (or higher) decrease in the encryption throughput. Therefore, Rijndael and Serpent can be implemented using almost the same amount of area as Twofish and RC6; but Twofish and RC6 can never reach the speeds of the fastest implementations of Rijndael and Serpent I8.

6. Implementation of the AES candidates in non-feedback cipher modes

6.1 *Choice of an architecture*

6.1.1 Alternative architectures

Traditional methodology for design of high-performance implementations of secret-key block ciphers, operating in non-feedback cipher modes is shown in Fig. 15. The basic iterative architecture, shown in Fig. 15a is implemented first, and its speed and area determined. Based on these estimations, the number of rounds K that can be unrolled without exceeding the available circuit area is found. The number of unrolled rounds, K, must be a divisor of the total number of cipher rounds, *#rounds*. If the available circuit area is not large enough to fit all cipher rounds, architecture with partial outer-round pipelining, shown in Fig. 15b, is applied. The difference between this architecture and the architecture with partial loop unrolling, shown in Fig. 5b, is the presence of registers inside of the combinational logic on the boundaries between any two subsequent cipher rounds. As a result, K blocks of data can be processed by the circuit at the same time, with each of these blocks stored in a different register at the end of a clock cycle. This technique of paralell processing multiple streams of data by the same circuit is called pipelining. The throughput and area of the circuit with partial outer-round pipelining increase proportionally to the value of K, as shown in Fig. 17, the encryption/decryption latency remains the same as in the basic iterative architecture, as shown in Fig. 18. If the available area is large enough to fit all cipher rounds, the feedback loop is not longer necessary, and full outer-round pipelining, shown in Fig. 15c, can be applied.

Our methodology for implementing non-feedback cipher modes is shown in Fig. 16. The primary difference is that before loop unrolling, the optimum number of pipeline registers is inserted inside of a cipher round, as shown in Fig. 16b. The entire round,
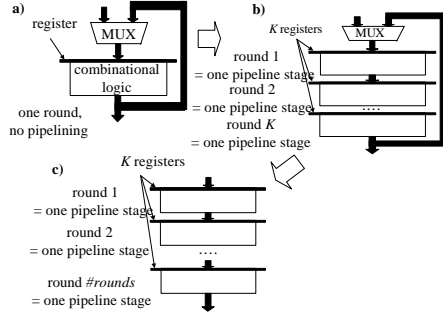
Fig. 15. Three architectures used traditionally to implement non-feedback cipher modes: a) basic iterative architecture, b) partial outer-round pipelining, c) full outer-round round pipelining.
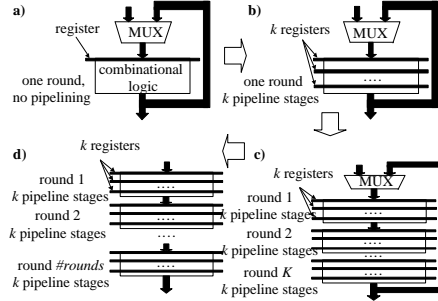

Fig. 16. Our architectures used to implement non-feedback cipher modes: a) basic iterative architecture, b) inner-round pipelining, c) partial mixed inner- and outer-pipelining, d) full mixed inner- and outer-round pipelining.

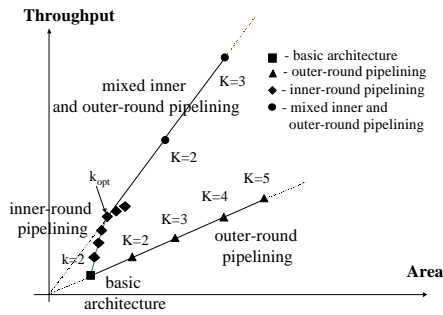**Fig. 17.** Throughput vs. area characteristics of alternative architectures working in feedback cipher modes.
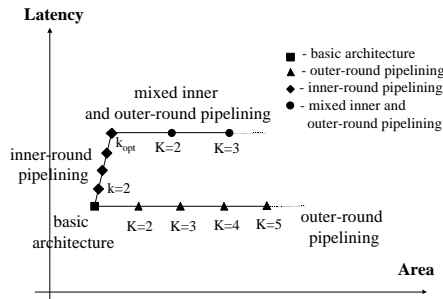

Fig. 18. Latency vs. area characteristics of alternative architectures working in feedback cipher modes.

including internal pipeline registers is than repeated K times (see Fig. 16c). The number of unrolled rounds K depends on the maximum available area or the maximum required throughput.

The primary advantage of our methodology is shown in Fig. 17. Inserting registers inside of a cipher round significantly increases cipher throughput at the cost of only marginal increase in the circuit area. As a result, the throughput to area ratio increases until the number of internal pipeline stages reaches its optimum value k_{opt}. Inserting additional registers may still increase the circuit throughput, but the throughput to area ratio will deteriorate. The throughput to area ratio remains unchanged during the subsequent loop unrolling. The throughput of the circuit is given by

$$Throughput\,(K, k) = K \cdot block\_size \,/\, \#rounds \cdot \mathrm{T_{CLKinner\_round}}\,(k) \qquad (4)$$

where k is the number of inner-round pipeline stages, K is the number of outer-round pipeline stages, and $\mathrm{T_{CLKinner\_round}}\,(k)$ is the clock period in the architecture with the k-stage inner-round pipelining.

For a given limit in the circuit area, mixed inner- and outer-round pipelining shown in Fig. 16c offers significantly higher throughput compared to the pure outer-round pipelining (see Fig. 17). When the limit on the circuit area is large enough, all rounds of the cipher can be unrolled, as shown in Fig. 16d, leading to the throughput given by

$$Throughput\ (\#rounds, k_{opt}) = block\_size\ /\ \mathrm{T_{CLKinner\_round}}\ (k_{opt}) \tag{5}$$

where k_{opt} is the number of inner-round pipeline stages optimum from the point of view of the throughput to area ratio.

The only side effect of our methodology is the increase in the encryption/decryption latency. This latency is given by

$$Latency(K, k) = \#rounds \cdot k \cdot \mathrm{T_{CLKinner\_round}}\ (k) \tag{6}$$

It does not depend on the number of rounds unrolled, K.

The increase in the encryption/decryption latency, typically in the range of single microseconds, usually does not have any major influence on the operation of the high-volume cryptographic system optimized for maximum throughput. This is particularly true for applications with a human operator present on at least one end of the secure communication channel.

6.1.2 Our choice

In our opinion, a fair methodology for comparing hardware performance of the AES candidates should fulfill the following requirements.

a) It should be based on the architecture that is likely to be used in practical implementations, because of the superior throughput/area ratio.

b) It should not favor any group of ciphers or a specific internal structure of a cipher.

For feedback cipher modes, both conditions are very well fulfilled by the basic iterative architecture, and this architecture was commonly used for comparison. For non-feedback cipher modes, the decisions about the choice of the architecture varied and no consensus was achieved.

The NSA team chose to use for comparison the full outer-round pipelining [7], [8]. In our opinion, this choice does not fulfill either one of the formulated above requirements. As shown in Fig. 17, the outer-round pipelining offers significantly worse throughput to area ratio compared to the architecture with the mixed inner- and outer-round pipelining. Therefore, the use of this architecture may lead to suboptimum designs, which are not likely to be used in practice. Secondly, the choice of the outer-round pipelining favors ciphers with a short and simple cipher round, such as Serpent and Rijndael. The AES candidates with more complex internal rounds, such as Mars, RC6, and Twofish, are adversely affected.

$$Throughput_{\text{full\_outer\_round}} = block\_size\ /\mathrm{T_{CLKbasic}} \tag{7}$$

where $\mathrm{T_{CLKbasic}}$ is a delay of a single round.

The throughput does not depend any longer on the number of cipher rounds, but is inversely proportional to the delay of a single round. Ciphers with the large number of simple rounds are favored over ciphers with the small number of complex rounds.

On the other hand, the throughput in the full mixed inner and outer-round pipelining is given by

$$Throughput_{full\_mixed} = block\_size\ /T_{CLKinner\_round}\ (k_{opt}) \qquad (8)$$

where $T_{CLKinner\_round}(k_{opt})$ is the delay of a single pipeline stage for the optimum number of registers introduced inside of a single round. In FPGA implementations, this delay is determined by the delay of a single CLB slice and delays of interconnects between CLBs. As a result, the throughput does not depend on the complexity of a cipher round and tend to be similar for all AES candidates. Based on these observations, we have decided that full mixed inner- and outer-round pipelining should be the architecture of choice for comparing hardware performance of the AES candidates in non-feedback cipher modes.

6.2 Our results and comparison with results of other groups

The results of our implementations of four AES candidates using full mixed inner- and outer-round pipelining and Virtex XCV-1000BG560-6 FPGA devices are summarized in Figs. 19, 21, and 22. Because of the timing constraints, we did not attempt to implement Mars in this architecture, nevertheless, we plan to pursue this project in the future. In Fig. 20, we provide for comparison the results of implementing all five AES finalists by the NSA group, using full outer-round pipelining and semi-custom ASICs based on the 0.5 μm CMOS MOSIS library [8].

To our best knowledge, the throughputs of the AES candidates obtained as a result of our design effort, and shown in Fig. 17, are the best ever reported, including both FPGA and ASIC technologies. Our designs outperform similar pipelined designs based on the use of identical FPGA devices, reported in [4], by a factor ranging from 3.5 for Serpent to 9.6 for Twofish. These differences may be attributed to using a suboptimum number of inner-round pipeline stages and to limiting designs to single-chip modules in [4]. Our designs outperform NSA ASIC designs in terms of the encryption/decryption throughput by a factor ranging from 2.1 for Serpent to 6.6 for Twofish (see Figs. 19 and 20). Since both groups obtained very similar values of throughputs for the basic iterative architecture (see Figs. 13 and 14), these large differences should be attributed primarily to the differences between the full mixed inner- and outer-round round architecture employed by our group and the full outer-round architecture used by the NSA team.

By comparing Figs. 19 and 20, it can be clearly seen that using full outer-round pipelining for comparison of the AES candidates favors ciphers with less complex cipher rounds. Twofish and RC6 are over two times slower than Rindael and Serpent I1, when full outer-round pipelining is used (Fig. 20); and have the throughput greater than Rijndael, and comparable to Serpent I1, when full mixed inner- and outer-round pipelining is applied (Fig. 19). Based on our basic iterative architecture implementation of Mars, we predict that the choice of the pipelined architecture would have the similar effect on Mars.

The deviations in the values of the AES candidate throughputs in full mixed inner- and outer-round pipelining do not exceed 20% of their mean value. The analysis of critical paths in our implementations has demonstrated that all critical

paths contain only a single level of CLBs and differ only in delays of programmable interconnects. Taking into account already small spread of the AES candidate throughputs and potential for further optimizations, we conclude that the demonstrated differences in throughput are not sufficient to favor any of the AES algorithms over the other. As a result, circuit area should be the primary criterion of comparison for our architecture and non-feedback cipher modes.

As shown in Fig. 21, Serpent and Twofish require almost identical area for their implementations based on full mixed inner- and outer-round pipelining. RC6 imposes over twice as large area requirements. Comparison of the area of Rijndael and other ciphers is made difficult by the use of dedicated memory blocks, Block SelectRAMs, to implement S-boxes. Block Select RAMs are not used in implementations of any of the remaining AES candidates, and we are not aware of any formula for expressing the area of Block Select RAMs in terms of the area used by CLB slices. Nevertheless,
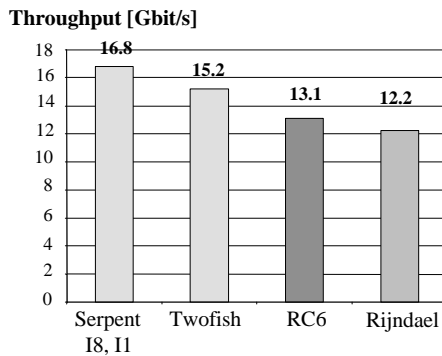
Throughput [Gbit/s]

Fig. 19. Full mixed inner- and outer-round pipelining, throughput for Virtex XCV-1000, our results.

Throughput [Gbit/s]
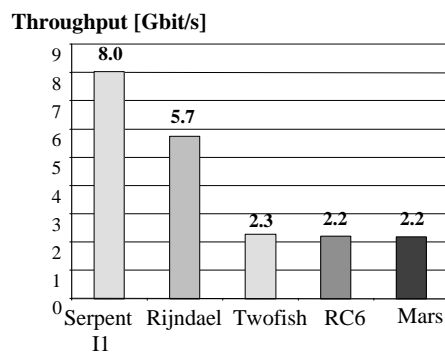
Fig. 20. Full outer-round pipelining, throughput for 0.5 μm CMOS standard-cell ASICs, NSA results.
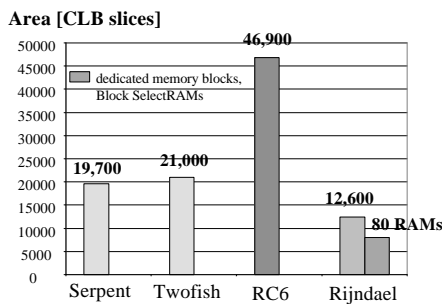
Area [CLB slices]

Fig. 21. Full mixed inner- and outer-round pipelining, area for Virtex XCV-1000, our results. Multiple XCV-1000 devices used when necessary.
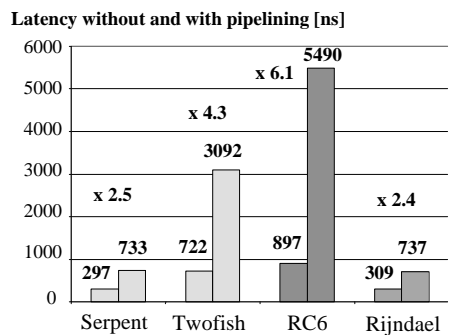
Latency without and with pipelining [ns]

Fig. 22. Increase in the encryption/decryption latency as a result of moving from the basic iterative architecture to full mixed inner- and outer-round pipelining. The upper number (after 'x') shows the ratio of latencies.

we have estimated that an equivalent implementation of Rijndael, composed of CLBs only, would take about 24,600 CLBs, which is only 17 and 25 percent more than implementations of Twofish and Serpent.

Additionally, Serpent, Twofish, and Rijndael all can be implemented using two FPGA devices XCV-1000; while RC6 requires four such devices. It should be noted that in our designs, all implemented circuits perform both encryption and decryption. This is in contrast with the designs reported in [4], where only encryption logic is implemented, and therefore a fully pipelined implementation of Serpent can be included in one FPGA device.

Connecting two or more Virtex FPGA devices into a multi-chip module working with the same clock frequency is possible because the FPGA system level clock can achieve rates up to 200 MHz [10], and the highest internal clock frequency required by the AES candidate implementation is 131 MHz for Serpent. New devices of the Virtex family, scheduled to be released in 2001, are likely to be capable of including full implementations of Serpent, Twofish, and Rijndael on a single integrated circuit.

In Fig. 22, we report the increase in the encryption/decryption latency resulting from using the inner-round pipelining with the number of stages optimum from the point of view of the throughput/area ratio. In majority of applications that require hardware-based high-speed encryption, the encryption/decryption throughput is a primary performance measure, and the latencies shown in Fig. 22 are fully acceptable. Therefore, in this type of applications, the only parameter that truly differentiates AES candidates, working in non-feedback cipher modes, is the area, and thus the cost, of implementations. As a result, in non-feedback cipher modes, Serpent, Twofish, and Rijndael offer very similar performance characteristics, while RC6 requires over twice as much area and twice as many Virtex XCV-1000 FPGA devices.

7. Summary

We have implemented all five final AES candidates in the basic iterative architecture, suitable for feedback cipher modes, using Xilinx Virtex XCV-1000 FPGA devices. For all five ciphers, we have obtained the best throughput/area ratio, compared to the results of other groups reported for FPGA devices. Additionally, we have implemented four AES algorithms using full mixed inner- and outer-round pipelining suitable for operation in non-feedback cipher modes. For all four ciphers, we have obtained throughputs in excess of 12 Gbit/s, the highest throughputs ever reported in the literature for hardware implementations of the AES candidates, taking into account both FPGA and ASIC implementations.

We have developed the consistent methodology for the fast implementation and fair comparison of the AES candidates in hardware. We have found out that the choice of an optimum architecture and a fair performance measure is different for feedback and non-feedback cipher modes.

For feedback cipher modes (CBC, CFB, OFB), the basic iterative architecture is the most appropriate for comparison and future implementations. The encryption/decryption throughput should be the primary criterion of comparison because it cannot be easily increased by using a different architecture, even at the cost

of a substantial increase in the circuit area. Serpent and Rijndael outperform three remaining AES candidates by at least a factor of two in both throughput and latency. Our results for feedback modes have been confirmed by two independent research groups.

For non-feedback cipher modes (ECB, counter mode), an architecture with full mixed inner- and outer-round pipelining is the most appropriate for comparison and future implementations. In this architecture, all AES candidates achieve approximately the same throughput. As a result, the implementation area should be the primary criteria of comparison. Implementations of Serpent, Twofish, and Rijndael consume approximately the same amount of FPGA resources; RC6 requires over twice as large area. Our approach to comparison of the AES candidates in non-feedback cipher modes is new and unique, and has yet to be followed, verified, and confirmed by other research groups.

Our analysis leads to the following ranking of the AES candidates in terms of the hardware efficiency: Rijndael and Serpent close first, followed in order by Twofish, RC6, and Mars. Combined with rankings of the AES candidates in terms of the remaining evaluation criteria, such as security, software efficiency, and flexibility, our study fully supports the choice of Rijndael as the new Advanced Encryption Standard.

References

1. "Advanced Encryption Standard Development Effort," http://www.nist.gov/aes.
2. *Third Advanced Encryption Standard (AES) Candidate Conference*, New York, April 13-14, 2000, http://csrc.nist.gov/encryption/aes/round2/conf3/aes3conf.htm.
3. J. Nechvatal, E. Barker, L. Bassham, W. Burr, M. Dworkin, J. Foti, and E. Roback, "Report on the Development of the Advanced Encryption Standard (AES)," available at [1].
4. A. J. Elbirt, W. Yip, B. Chetwynd, C. Paar, "An FPGA implementation and performance evaluation of the AES block cipher candidate algorithm finalists," in [2].
5. A. Dandalis, V. K. Prasanna, J. D. Rolim, "A Comparative Study of Performance of AES Final Candidates Using FPGAs," *Proc. Cryptographic Hardware and Embedded Systems Workshop*, CHES 2000, Worcester, MA, Aug 17-18, 2000.
6. N. Weaver, J. Wawrzynek, "A comparison of the AES candidates amenability to FPGA Implementation," in [2].
7. B. Weeks, M. Bean, T. Rozylowicz, C. Ficke, "Hardware performance simulations of Round 2 Advanced Encryption Standard algorithms," in [2].
8. B. Weeks, M. Bean, T. Rozylowicz, C. Ficke, "Hardware performance simulations of Round 2 Advanced Encryption Standard algorithms," NSA's final report on hardware evaluations published May 15, 2000, available at http://csrc.nist.gov/encryption/aes/round2/r2anlsys.htm#NSA.
9. T. Ichikawa, T. Kasuya, M. Matsui, "Hardware Evaluation of the AES Finalists," in [2].
10. Xilinx, Inc., "Virtex 2.5 V Field Programmable Gate Arrays," available at http://www.xilinx.com.
11. National Security Agency, "Initial plans for estimating the hardware performance of AES submissions," available at http://csrc.nist.gov/encryption/aes/round2/round2.htm
12. *Symmetric Key Block Cipher Modes of Operation Workshop*, Baltimore, October 20, 2000, available at http://csrc.nist.gov/encryption/aes/modes/