Wireless Personal Communications (2022) 127:3615–3635
https://doi.org/10.1007/s11277-022-09933-2

Performance Analysis of Enhanced AES-128 and Blowfish Algorithms Through Parallel-Pipelined-Memory Techniques

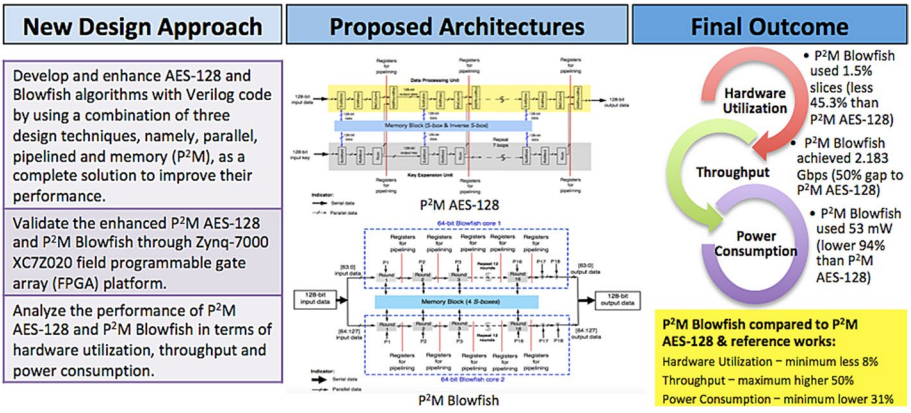
Rafidah Ahmad[1] · Mohamad Faiz Mohamed Omar[1] · Jagadheswaran Rajendran[1] · Widad Ismail[2]

Accepted: 1 July 2022 / Published online: 20 July 2022

Abstract

Currently, the advanced encryption standard (AES)-128 algorithm is deployed by the Institute of Electrical and Electronics Engineers standards, and it is widely used to secure wireless communication in various radio frequency bands. This paper proposes an enhanced Blowfish algorithm with better performance than the AES-128 as a potential security function to be implemented in mobile devices. A performance analysis based on Artix-7 field programmable gate array platform is conducted in terms of the design throughput, hardware utilisation and power consumption of the proposed AES-128 and Blowfish architectures, which are enhanced by using the parallel–pipelined–memory (P^2M) techniques. These techniques contribute to the performance improvement of the P^2M AES-128 and P^2M Blowfish, with the proposed Blowfish achieving 50% better power throughput and 45.3% lower logic resources. Findings show that the P^2M Blowfish not only secures mobile devices but can also reduce the design space and prolong battery lifetime for longer usage at a high data rate.

Graphical abstract



Keywords AES-128 · Blowfish · Parallel-pipelined-memory · FPGA · Power-throughput

Extended author information available on the last page of the article

1 Introduction

The advanced encryption standard (AES)-128 and Blowfish algorithms are both from the symmetric-key block cipher cryptography family. The AES scheme is used by most of the Institute of Electrical and Electronics Engineers (IEEE) standards to secure wireless communication amongst mobile devices. The AES-128 scheme can process 128-bit data blocks with 128-bit cipher keys. Its plain text goes through 10 rounds (N_r), 4 columns (N_b) and 4 cipher keys (N_k). Each encryption round performs different operations, such as byte substitution (*SubBytes*), shift rows (*ShiftRows*), mix-column (*MixColumns*) and addition of a round key (*AddRoundKey*). Meanwhile, the decryption round performs inverse transformations, such as *InvSubBytes*, *InvShiftRows*, *InvMixColumns* and *AddRoundKey*. However, this scheme requires high computing platform and large design size because of its complex architecture. A few attacks, such as related-key attack and side-channel attack on its safety level [1–4], which can cause doubt among users or providers, against AES have also been found. Therefore, the issues that should be considered are performance and security because mobile devices have limited battery power and are designed to be portable with lots of features [5, 6]. Most of the current research trends are more concerned with simple and high-speed security architecture [5, 6].

To overcome this issue, an alternative security scheme with improved power throughput and low hardware utilisation is introduced and developed in this paper. On the basis of performance analysis between AES and Blowfish schemes in previous works [7–16], Blowfish is considered the alternative scheme to replace existing AES because it has better performance, simpler architecture and high security. On the basis of [17], Blowfish has a 64-bit block size and a variable key length from 32 bits up to 448 bits. The Blowfish scheme consists of two units, namely, key expansion and data encryption units. The 64-bit input data of this scheme are divided into two 32-bit halves, and the *P-Arrays* (P1-P18), which comprise 18 32-bit subkeys for the key expansion unit, are used. This scheme has 16 rounds, with each round implementing the Feistel (F) function. In the F function block, four 32-bit *S-boxes* have 256 entries each. After the 16th round, two 32-bit half data are recombined to obtain the cipher text during the encryption mode. As for the decryption process, the Blowfish flow refers to the inverse of the encryption process. All operations involve only XORs and additions (ADDs) of 32-bit data. On the basis of [2, 16, 18], Blowfish is proven to be a highly secure and strong encryption scheme. Furthermore, in the proposed research, the Blowfish is designed and executed for 16 rounds to enable high security encryption [4]. Given that Blowfish is also unpatented and freely available, it may have the potential to replace existing AES to achieve a high-performance end product with better security level.

This work proposes the development of enhanced AES-128 and Blowfish algorithms with Verilog code by using a combination of three design techniques, namely, parallel, pipelined and memory (P^2M), as a complete solution instead of applying a segregated approach; these techniques are the main contributions of this study. Through Zynq-7000 XC7Z020 field programmable gate array (FPGA) platform with Artix-7 technology, the proposed P^2M AES-128 and P^2M Blowfish are implemented as a prototyping product to verify their functionality and complexity in real-time environments. Subsequently, the performance of the proposed AES-128 and Blowfish is analysed in terms of design throughput, logic resource and power consumption as another contribution of this study. On the basis of the performance results, this study can guide researchers to determine the possibility of designing the proposed P^2M Blowfish through application-specific integrated circuit (ASIC) methodology to produce chipsets before being implemented in mobile devices for

secure wireless communication instead of the AES-128. This work supports the current research trends that focus on developing simple and high-speed security architecture via the P^{2}M Blowfish since mobile devices have limited core storage and power source.

This paper is organised as follows. Section 2 discusses the related research on the AES-128 and Blowfish designs through FPGA platforms. Section 3 describes the design methodology of the proposed AES-128 and Blowfish architectures by using parallel, pipelined and memory techniques. Section 4 explains their results and discussion in terms of FPGA hardware utilisation, throughput and power consumption. Section 5 concludes this research.

2 Related Research

Studies that conduct performance analysis on the AES-128 and Blowfish designs based on FPGA platforms are limited. Table 1 shows that previous researchers used different design techniques either in the data processing unit or key data processing unit of their AES-128 architectures. The sequential technique in [19–21] is known as the conventional method because it involves only the consecutive data signal flows of the design circuit, which could increase the clock cycles. Then, the memory technique was used by Toubal et al. [22] to store the keys generated during the encryption process. However, other related works in [23–31] showed that instead of using the memory block, the registers were used to store a large amount of data of *S-boxes* and inverse *S-boxes* for encryption and decryption, respectively. The findings in Table 1 show that the highest throughput of 1.085 Gbps and lowest power consumption of 0.88 W were achieved by the proposed P^{2}M AES-128 if compared to the reference works. Nuray et al. [28] obtained the least logic resources, as indicated by the 1% usage of slices through pipelined and parallel techniques in their AES-128 architecture. These findings also show that the AES-128 designs obtained better performance in terms of slices used, throughput and power consumption with the memory, pipelined or parallel technique compared with the sequential technique. However, in reality, these results have proven the difficulty in obtaining the best performance in all parameters at once. Thus, appropriate design techniques must be implemented in the AES architecture by considering many issues, such as time constraint, core density and power for the design activities, to achieve the best performance as much as possible.

The performance of 64-bit Blowfish designs from previous research is summarised in Table 2. Previous works used either a single design technique or a combination of two design techniques that comprises sequential, parallel, pipelined or memory techniques in their Blowfish architectures. The sequential, parallel and pipelined techniques were mostly used to design the sub-blocks of data processing and key processing units. Meanwhile, the memory technique was used to store a large amount of data of four *S-boxes* and *P-Arrays*. The performance analysis indicates that the Blowfish designed by Sudarshan et al. [32] using pipelined and memory-based techniques has the smallest core size, using only 214 slices. Nalawade and Gawali [39] achieved the highest throughput of 1.632 Gbps using the memory technique in their Blowfish architecture. However, the Blowfish designed by Karthigaikumar and Baskaran [33], which used pipelined and parallel techniques, is the best design for power saving in mobile devices because their Blowfish consumed only 77 mW power.

Although the AES-128 and Blowfish designs in previous works were developed on different FPGA families, this is not an issue because the FPGA is used only as a

Table 1 Performance analysis on AES-128 designs from previous research and proposed work

References	FPGA family	Design technique	Slices used	Clock frequency (MHz)	Throughput (Gbps)	Power consumption (mW)
Fan and Hwang [19]	Virtex2 XC2V3000-6	Sequential, parallel and pipelined	7617/14336 (53%)	75.3	0.876	NA
Prasanthi and Reddy [20]	Virtex XCV600BG560-6	Sequential and pipelined	1853/6912 (26%)	140.4	0.352	NA
Subhashini and Jagadeeswari [21]	Spartan6	Sequential	2983/51840 (6%)	NA	NA	3516
Toubal et al. [22]	Artix-7 XC7A35T	Pipelined and memory	3054/20800 (14.7%)—(LUT)	150.0	0.295	NA
Elbirt et al. [23]	Virtex XCV1000BG560-4	Pipelined	10,286/12288 (84%)	5.6	0.237	NA
Yoo et al. [24]	Virtex2 XC2VP70-7	Pipelined	11,433/33088 (35%)	125.6	0.268	2083
Adib et al. [25]	Virtex5 XC5VLX50	Parallel and pipelined	587/7200 (8%)	346.2	0.426	NA
Wang and Ha [26]	Virtex6 XC6VLX240T	Pipelined	10,155/37680 (30%)	302.2	0.586	9410
Neenu and Bonifus [27]	Spartan XC3S100E5VQ100	Parallel	930/960 (97%)	81.7	0.654	NA
Nuray et al. [28]	Virtex6 XC6VLX75T-2	Pipelined and parallel	169/11640 (1%)	393.0	0.217	NA
Guruprasad and Chandrasekar [29]	Artix-7 XC7A100TCSG324	Pipelined	989/15850 (6%)	291.6	0.888	NA
Nabil et al. [30]	Spartan 3A/3AN starter kit*	Pipelined and parallel	46,745[a], 56,845[b]	NA	0.75 cycles/byte[a], 0.125 cycles/byte[b]	NA
Kumar et al. [31]	Virtex5*	Pipelined	4879/7200 (68%)	733.2	NA	NA
Proposed P^{2}M AES-128	Zynq-7000 XC7Z020	Parallel, pipelined and memory	6231/13300 (47%)	250	1.085	880

NA, Not available

*The specific FPGA family was not mentioned by the authors

[a]pipelined

[b]parallel

Table 2 Performance analysis on 64-bit Blowfish designs from previous research

Ref	FPGA family	Design technique	Slices used	Clock frequency (MHz)	Throughput (Gbps)	Power consumption (mW)
Sudarshan et al. [32]	Spartan2E 2S300EFG456-6	Pipelined and memory	214/3072 (7%)	73.8	0.779	NA
Karthigaikumar and Baskaran [33]	Virtex2 XCV50BG256-6	Pipelined and parallel	1608/1728 (93%)	167.0	0.563	77
Kurniawan et al. [34]	Virtex4 XC4VLX25-SF363	Sequential	678/10752 (6%)	NA	NA	NA
Oukili and Bri [35]	Virtex5 XC5VLX220T-2FF1738	Pipelined and parallel	1280/34560 (3%)	187.6	0.353	NA
Suresh and Neema [36]	Virtex5 XC5VLX50T	Parallel	9971/28800 (35%)	NA	0.019	582
Bansal and Jassal [37]	Virtex4 XC4VLX25-SF363	Sequential and memory	574/10752 (5%)	133.2	NA	NA
Chatterjee et al. [38]	Spartan3E XC3S500E-5FG320	Parallel and pipelined	3222/4656 (69%)	295.6	0.386	NA
Nalawade and Gawali [39]	Virtex5 XC5VLX50T	Memory	420/28800 (1%)	153.0	1.632	460

NA, Not available

medium to verify the functionality and complexity of the cryptography architectures. The implementation of AES and Blowfish designs on different FPGA platforms is also considered a benchmark to represent the performance analysis among reference works. Therefore, Tables 1 and 2 indicate that the combination of at least two design techniques in each cryptography architecture, which consist of pipelined, parallel or memory, could provide support in achieving either the lowest hardware utilisation, lowest power consumption or highest throughput. The performance analysis also showed that the weakest performance of AES-128 and Blowfish designs was obtained by using one of the pipelined, parallel or sequential techniques in its architecture. However, none of the reference works on the AES-128 and Blowfish designs achieved the best performance all at once in terms of hardware utilisation, throughput and power consumption despite employing different techniques. These outcomes confirm the possibility of designing the AES-128 and Blowfish architectures with the combination of three techniques, namely, parallel, pipelined and memory techniques, to improve their performance results further as a contribution of this study.

3 Design Methodology

With the use of Xilinx Vivado version 2015.2, the proposed P^{2}M AES-128 and P^{2}M Blowfish were designed using the hardware description language code called Verilog. Both cryptography designs have a 128-bit block size and 128-bit key length to achieve a fair performance comparison. The design methodology for the architectures of the proposed AES-128 and Blowfish is explained in Sect. 3.1 and 3.2, respectively. After this step, both the Verilog codes of these architectures are implemented on the Zynq-7000 FPGA platform for hardware verification, as shown in Fig. 1. This process begins by generating a bit stream file via Xilinx software, which contains a binary sequence of each proposed AES-128 and Blowfish. These files are downloaded on the FPGA platform by controlling the input data and clock frequency with the use of a logic analyser and signal generator, respectively. The output data are also monitored through the logic analyser. The three parameters of performance analysis, namely, hardware utilisation, throughput and power consumption, are determined by using the Xilinx FPGA software.

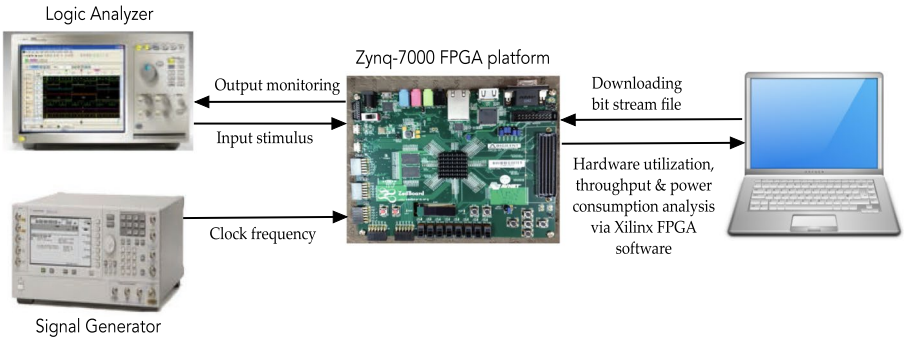


Fig. 1 Implementation setup for the proposed P^{2}M AES-128 and P^{2}M Blowfish

3.1 P²M AES-128

The proposed AES-128 design comprises two important units, namely, data processing and key expansion. The data processing unit for encryption consists of four main transformations, namely, *SubBytes*, *ShiftRows*, *MixColumns* and *AddRoundKey*. These transformations can be inverted and implemented in reverse order to produce a decryption function. The inverse transformations include *InvSubBytes*, *InvShiftRows*, *InvMixColumns* and *AddRoundKey*. The proposed AES-128 is designed by using the parallel, pipelined and memory techniques to improve the power throughput with reduced hardware utilisation. Designing the AES-128 architecture using these techniques is part of the contribution of this study. The difference between the conventional AES-128 architecture by using sequential technique and the proposed P²M AES-128 architecture is illustrated in Fig. 2. As shown in Fig. 2a, the sequential technique, which was implemented by Subhashini and Jagadeeswari [21], executed the data signal only in serial order and then transformed the data consecutively in every round. Instead of memory block, the registers were used to store a large amount of data of *S-box* and inverse *S-box*. This technique could increase the employment of flip-flops (FFs) and slow down the speed of AES-128 performance because each register had its own timing delay. The power consumed by the conventional design would also be increased.

Unlike the proposed P²M AES-128 based on Fig. 2b, parallel technique is used to execute 128-bit input data and 128-bit input key data at every round to obtain the data from the memory of *S-box* and inverse *S-box* for the processes of *SubBytes* and *SubWord* in the data processing and key expansion units, respectively. A total of 256 entries of 8-bit *S-box* or inverse *S-box* values are stored in a lookup table (LUT)-based random access memory (RAM) block. Figure 3 shows additional details on the data path of *SubBytes* and *SubWord* transformations with the implementation of parallel and memory techniques. These transformations were continuously repeated 10 times for every 128-bit data frame. The *S-box* is used when the *mode* is '1' for the encryption process, and the inverse *S-box* is used when *mode* is '0' for the decryption process. Through this technique, the execution time for *SubBytes* and *SubWord* transformations can be accelerated to obtain a high design throughput. The hardware requirement for these transformations can also be reduced because the same memory is shared, thereby resulting in low power consumption.

In this research, the pipelined technique is implemented to achieve the highest possible throughput by dividing AES-128 design into partitions and by placing registers. The registers comprised FFs with a reset function. Figure 2b shows the data path of pipelining in the P²M AES-128 architecture. Every output port of *MixColumn* and *Rcon* is also defined as a register at every round to reduce many critical paths and synchronise the data. In the final round, the 128-bit cipher text is obtained after the encryption, or the 128-bit original text is regained after the decryption.

3.2 P²M Blowfish

The P²M Blowfish design comprises two important units: data processing and key expansion units. On the basis of [17], the data processing unit employs 18 *P-Arrays* and four *S-boxes* in the F function for encryption or decryption execution within 16 rounds, as shown in Algorithm 1. The *P-Array* consists of 32-bit subkeys, which are generated in the key expansion unit, as depicted in Algorithm 2 based on [17]. Both

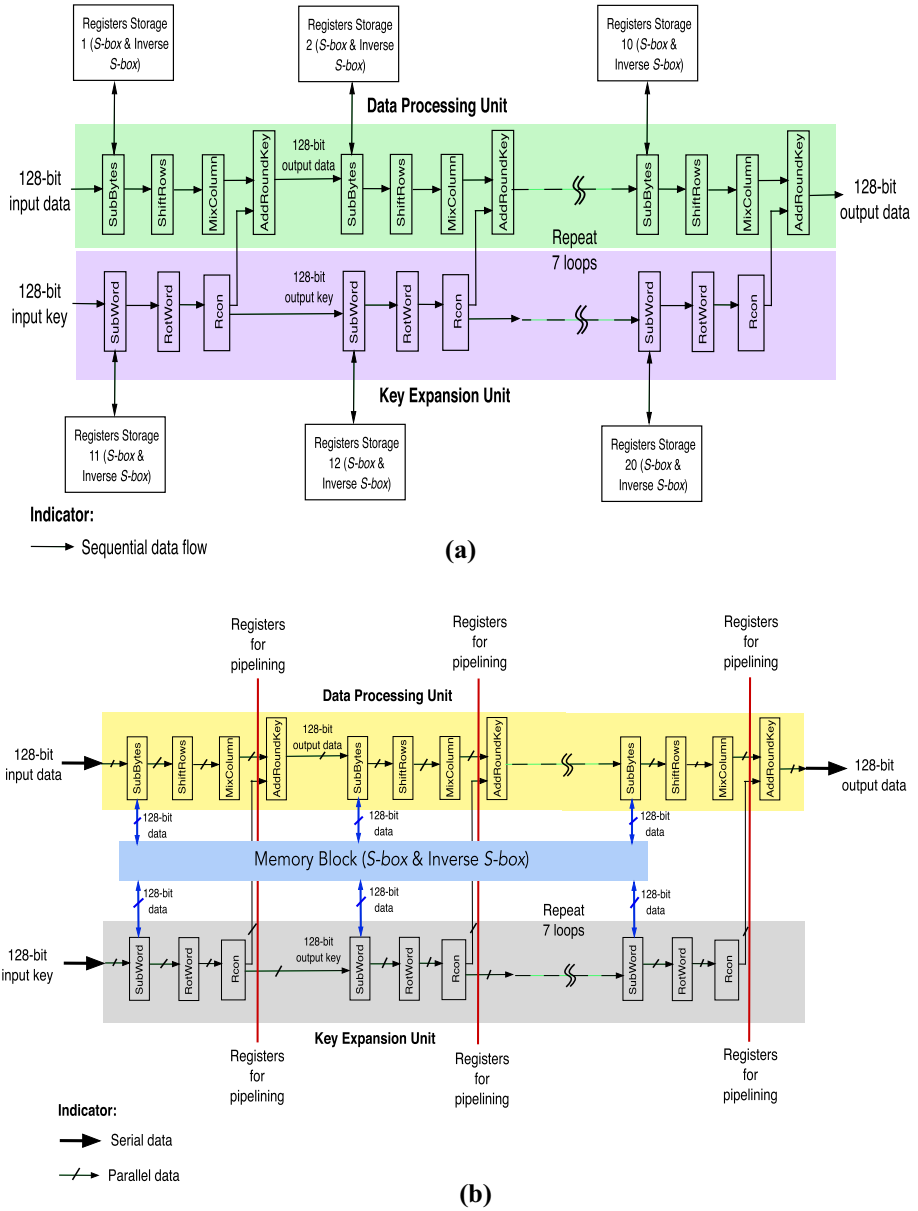


Fig. 2 Differences in AES-128 architectures: **a** Conventional methodology; **b** Proposed P^{2}M methodology

units involve only the XOR and ADD operations. As part of the contribution of this study, a combination of parallel, pipelined and memory techniques was used to increase the P^{2}M Blowfish design throughput and reduce its hardware utilisation and power consumption. In general, Fig. 4 illustrates the difference between the conventional 64-bit Blowfish architecture by using the sequential technique and the proposed P^{2}M Blowfish architecture. As shown in Fig. 4a, Kurniawan et al. [34] used the sequential technique in

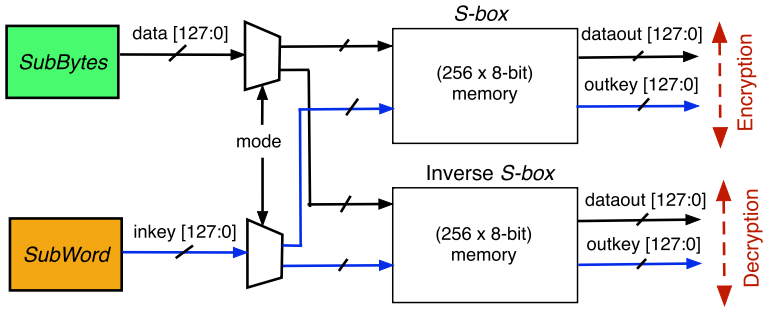
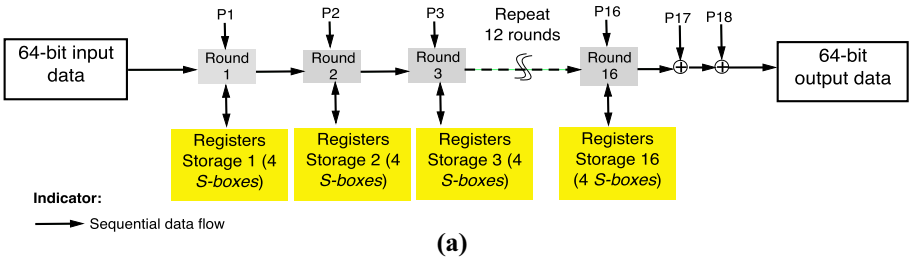
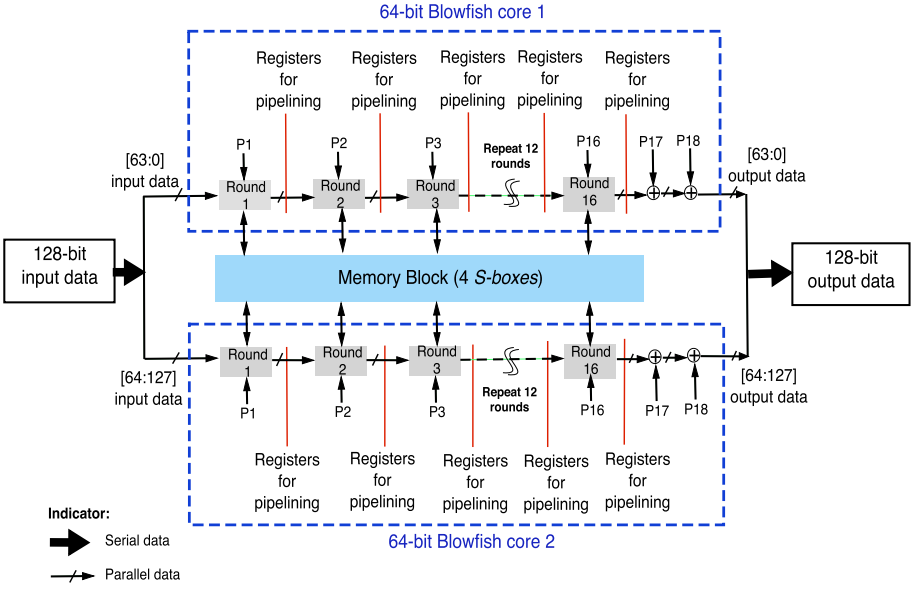


Fig. 3 Parallel and memory-based techniques in the proposed P²M AES-128 architecture



(a)



(b)

Fig. 4 Differences in Blowfish architectures **a** conventional methodology; **b** proposed P²M methodology

their Blowfish design, where the data signal was executed and processed consecutively in every round. The registers were used to store a large amount of data of four *S-boxes*. This factor could increase the logic resources and power consumption, and decrease the speed of their Blowfish performance.

1. 64-bit input data (x) divided into two 32-bit halves as xL and xR
2. 32-bit $xL = xL \hat{} Pi$ with $1 \leq i \leq 16$
3. 32-bit xR = F function $xL \hat{} xR$
   For F function
   - 32-bit xL divided into four 8-bit quarters as input data to four *S-boxes* (*S1, S2, S3, S4*)
   - Value of 8-bit input data of each *S-box* assigned to the value of 32-bit data from the memory as the output data
   - F function $xL = ((S1+S2) \hat{} S3)+S4$
   End for
4. 32-bit xL and 32-bit xR are swapped
5. Repeat steps 1 to 4 for 16 rounds
6. After 16th round, xL and xR are swapped again
7. $xR = xR \hat{} P17$
8. $xL = xL \hat{} P18$
9. 64-bit output data = $xL + xR$

Algorithm 1. 64-bit Blowfish operation for Verilog coding

1. 18 *P-Arrays* (*P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15,P16,P17,P18*) and four *S-boxes* (*S1,S2,S3,S4*) are initialised in order with a fixed string (hexadecimal digits of π)
2. *P1*^first 32-bit input key data; *P2*^second 32-bit input key data; repeat the same process until *P18*
3. All-zero strings are encrypted with the Blowfish operation by using the subkeys from steps 1 and 2
4. *P1* and *P2* are replaced with the output of step 3
5. Outputs of step 3 are encrypted with the Blowfish operation by using the modified subkeys
6. *P3* and *P4* are replaced with the output of step 5
7. Continue replacing all entries of the *P-Arrays* and *S-boxes* in order with the output of the continuously changing Blowfish operation

Algorithm 2. Subkeys generation for Verilog coding

In the P^{2}M Blowfish architecture, the parallel technique is used to combine the two 64-bit Blowfish cores to obtain the Blowfish of 128-bit block size for a fair performance comparison with the P^{2}M AES-128. On the basis of Fig. 4b, the two Blowfish cores contain the standard Blowfish operation. Both Blowfish cores are executed concurrently as dual-core and they share the same memory block, which is used to store the data of 18 *P-Arrays* and four *S-boxes,* which are presented in hexadecimal form. The pipelined technique was employed in every round to increase the design throughput and ensure accurate

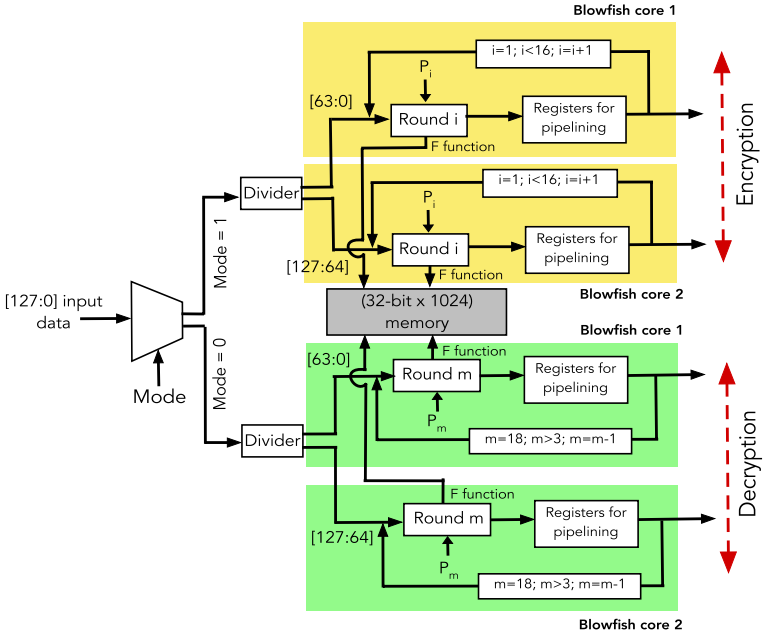


Fig. 5 Parallel, pipelined and memory-based techniques in the proposed 128-bit Blowfish architecture

timing for real-time communication. In the first round of both the parallel 64-bit Blowfish cores, pipelining path begins with every 64-bit output data after the F function are stored in the registers for the Blowfish operation in the next round. The F function comprises of four 32-bit *S-boxes* that are processed by using the XOR and ADD operations for the encryption or decryption process within 16 rounds. At the last two rounds, the 64-bit data of each Blowfish core are only swapped and operated with the XOR function before being concatenated to obtain 128-bit final output data.

Specifically, the BRAM of 32-bit with 1024 entries of π data as shown in Fig. 5 is deployed for F function in the proposed Blowfish architecture. Through memory technique, the usage of registers can be decreased which can help speed up the execution time of the Blowfish encryption or decryption process. Basically, the *mode* is used to select the encryption process at logic '1' or decryption process at logic '0'. Then, the input data of 128-bit are divided into two 64-bit data for execution of Blowfish algorithm simultaneously with F function in each round. The F function in both the Blowfish cores shared the same memory block and operated in parallel. The generated output data in each round are stored in the registers for pipelining purpose.

4 Results and Discussion

Another contribution of this study is that the proposed P^2M AES-128 and P^2M Blowfish were synthesised and implemented on Xilinx Zynq-7000 XC7Z020 FPGA core with Artix-7 technology to analyse their performances in terms of three parameters, namely, hardware utilisation, throughput and power consumption. The maximum clock frequencies

Table 3 The value of t_{hd} and t_{su} at different clock frequency

Proposed design	Clock frequency (MHz)	t_{hd} (ns)	t_{su} (ns)
P^{2}M AES-128	100	0.091	4.081
	150	0.089	3.170
	200	0.084	2.821
	250*	0.071	2.243
	251	0.062	−0.105
P^{2}M Blowfish	100	0.076	3.653
	150	0.074	3.233
	200	0.087	1.976
	250	0.081	0.862
	300	0.074	0.430
	324*	0.076	0.092
	325	0.071	−0.079

*Maximum

of P^{2}M AES-128 and P^{2}M Blowfish are 250 and 324 MHz, respectively. These maximum clock frequencies are obtained at the limitation of the data rate before the simulation waveform began to have a timing error. Generated from the Xilinx software, Table 3 shows the list of hold time, t_{hd} and setup time, t_{su} at certain clock frequency for both the proposed AES-128 and Blowfish. The setup time limits the fastest frequency for the clock which means the shortest period of data signal and hold time must be met to have proper operation [40]. With the two design techniques, the maximum clock frequency of the proposed AES-128 and Blowfish could be increased to meet the hold time and lead to a higher throughput. The timing analysis from Table 3 can also be used as a guideline to identify the maximum clock frequency of the enhanced P^{2}M AES-128 and P^{2}M Blowfish.

The performance results of these cryptography designs are analysed by using the Xilinx software and discussed as follows.

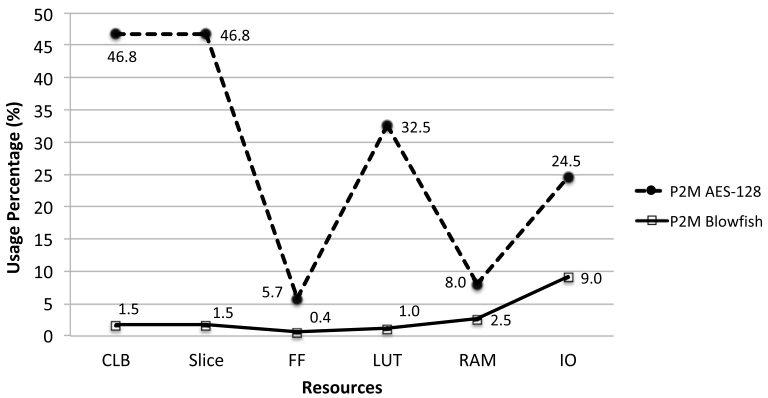


Fig. 6 Hardware utilisation between the proposed P^{2}M AES-128 and P^{2}M Blowfish

4.1 Hardware Utilization

The FPGA hardware utilised by the proposed P^{2}M AES-128 and P^{2}M Blowfish is depicted in Fig. 6. The generated implementation report from Xilinx software shows that the proposed Blowfish core is smaller, with 45.3% less usage of the configurable logic block (CLB) and slices compared with the proposed AES-128 core. On the basis of [41], a CLB is formed by two slices, which comprise the LUTs and FFs. All the logic functions of both the proposed AES-128 and Blowfish are operated here. The finding also shows that the proposed Blowfish design used 31.5% less LUT and 5.3% less FFs than the one required by the proposed AES-128. A larger memory is needed by the proposed AES-128 for *S-box* data storage with a difference of 5.5% if compared with the proposed Blowfish. Only 9% of the input–output (IO) block is used by the proposed Blowfish for real-time implementation through the Zynq-7000 platform. The use of the parallel, pipelined and memory techniques in both proposed cryptography architectures has more impact on the FPGA resources of the proposed P^{2}M Blowfish core. This result also confirms that the proposed P^{2}M Blowfish operation is less complex and has a smaller core size than the proposed P^{2}M AES-128. This characteristic proves that the proposed P^{2}M Blowfish core is more suitable to be implemented in wireless mobile devices with compact functions and low cost for secure communication.

4.2 Throughput

In this work, throughput was directed to evaluate the characteristic of the proposed cryptography architecture and its performance on Zynq-7000 FPGA. Throughput was calculated by using Eq. (1) based on [23, 30], where it involves the design data size in bits at a maximum frequency within the encryption or decryption latency. Latency is the time interval between the start of encryption or decryption of per block data and the start of the output data, where the encryption or decryption process of the proposed AES-128 and Blowfish includes the execution time of data and key expansion operations. Latency is calculated in clock cycles [23, 30].

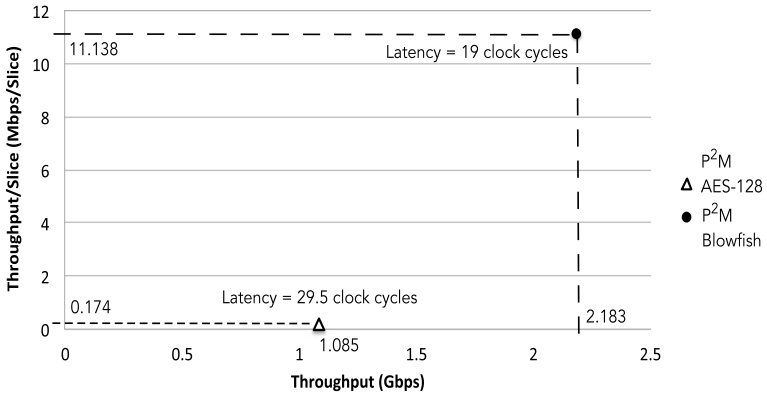


Fig. 7 Performance data of the proposed P^{2}M Blowfish and P^{2}M AES-128 at maximum clock frequency

$$\text{Throughput(Gbps)} = \frac{\text{Data Size(bits)} * \text{MaximumClockFrequency(MHz)}}{\text{Latency}} \quad (1)$$

Throughput per slice can be compared by using data transmission speed and design size. This procedure is the most objective method of comparing different security architectures on an FPGA device [42]. The equation for throughput per slice is shown below.

$$\text{Throughput/slice} = \frac{\text{Throughput(Gbps)}}{\text{No.ofslicesused}} \quad (2)$$

Figure 7 shows that the throughput of the proposed Blowfish is higher than that of the proposed AES-128 with a 50% gap at a latency of 19 clock cycles. Meanwhile, the throughput per slice for the proposed Blowfish is 98% higher than that of the proposed AES-128. This result shows that with a small design core, the proposed Blowfish can encrypt and decrypt data faster by using the parallel, pipelined and memory techniques.

4.3 Power Consumption

The Vivado Power tool from Xilinx software was used to analyse the power consumption through its power report. In this research, only the dynamic power was analysed during the implementation stage to provide the most accurate power estimation of the user design [43]. This choice was made because the netlist optimisation that affects the final logic resource utilisation, such as register replication or retiming, was taken into account. By default, implementation tools aim to achieve the design performance objective and minimise device utilisation. This idea means that the use of small FPGA hardware corresponds to a low consumption of the dynamic power. Dynamic power is associated with user design activity and switching events in the core or IO of the device [43]. This power depends on the voltage level, logic and routing resources used by the user design and determined as Eq. (3) [43].

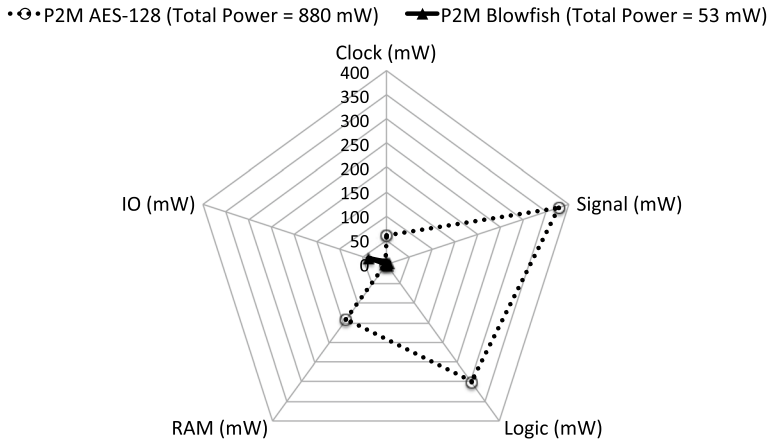


Fig. 8 Power consumption of the proposed P^{2}M AES-128 and P^{2}M Blowfish at 100 MHz clock frequency

$$\text{Dynamicpower(W)} = (\text{Clock} + \text{Logic} + \text{Signals} + \text{BRAM} + \text{IO})(\text{W}) \quad (3)$$

On the basis of Fig. 8, the power analysis shows that the proposed P^{2}M Blowfish has a total power consumption of 53 mW, which is a 94% difference from that of the proposed AES-128. This analysis is conducted at 100 MHz clock frequency for both the proposed cryptography designs as the benchmark for their power comparison. Given the simpler function of the proposed P^{2}M Blowfish than that of the P^{2}M AES-128, the lowest power consumption can be achieved through the employment of parallel, pipelined and memory techniques in its architecture. This characteristic can help prolong the battery lifetime of mobile devices which can reduced its operation cost while the security function is executed.

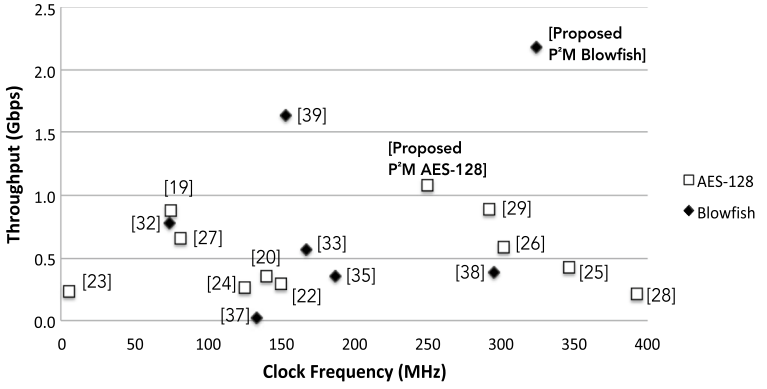


Fig. 9 Performance comparison between the proposed P^{2}M AES-128 and P^{2}M Blowfish with the reference works: **a** throughput vs. clock frequency; **b** power consumption vs. slices

4.4 Performance Comparison with Reference Works

The performance of the proposed P^{2}M AES-128 and P^{2}M Blowfish is compared with that of reference works based on the FPGA platform in terms of throughput, power consumption and hardware utilisation. These comparisons can be considered a guideline for researchers to evaluate the performance improvement that was achieved by the proposed AES-128 and Blowfish through parallel, pipelined and memory techniques [42]. As depicted in Fig. 9a, at 250 MHz clock frequency, the throughput of the proposed P^{2}M AES-128 is the highest among the previous AES-128 designs with at least an 18% gap. The proposed P^{2}M Blowfish achieved the highest throughput among others with a maximum difference of 50% at 324 MHz clock frequency.

Figure 9b shows that the proposed P^{2}M Blowfish requires only 8% slices less than the one in [32] with the lowest power consumption of a minimum of 31% compared with the others, thus being the best-enhanced cryptography design. The proposed P^{2}M AES-128 has lower power consumption with a minimum gap of 58% and the use of 6231 slices compared with previous AES-128 designs. These results prove that the use of the parallel, pipelined and memory techniques could shorten the latency of the proposed cryptography designs to speed up their execution time. The combination of these techniques can also reduce the use of slices, which represents hardware utilisation and contributes to low power consumption [42]. Overall, the characteristics of P^{2}M Blowfish can lead to a longer battery lifetime with a small core space of mobile devices and effective cost at a high data speed while performing security function.

5 Conclusions

In this work, the AES-128 and Blowfish algorithms were enhanced by using three design techniques, namely, parallel, pipelined and memory techniques. The performance of the proposed P^{2}M AES-128 and P^{2}M Blowfish in terms of design throughput, hardware utilisation and power consumption was analysed. The findings show that the P^{2}M Blowfish performed the best with at least an 8% difference compared with the P^{2}M AES-128 and other reference works. These performance results also prove that the proposed P^{2}M Blowfish has a possibility to replace the AES-128 as an existing cryptography algorithm, which is still being employed in mobile devices according to the IEEE standards. With its small design core, high throughput and low power consumption, the P^{2}M Blowfish is suitable for use in mobile devices at low cost as a security feature to support wireless communication. In future works, the proposed AES-128 and Blowfish designs will be designed by using complementary metal oxide semiconductor 0.18 μm technology via ASIC methodology for further performance analysis.

Authors' Contributions Conceptualization: RA, WI; Methodology: RA; Formal analysis and investigation: RA, MFMO; Writing—original draft preparation: RA; Writing—review and editing: JR, WI; Resources: RA; Supervision: WI.

Funding This research was financially supported by the Universiti Sains Malaysia Short Term Research Grant (Project No. 304/PCEDEC/6315464).

Data Availability Not applicable.

Code availability Not applicable.

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

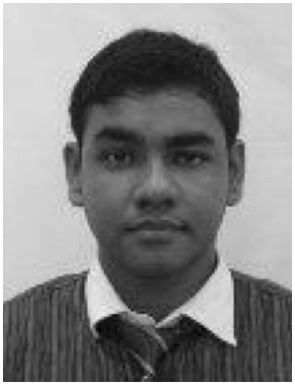
References

1. Niu, Y., Zhang, J., Wang, A., & Chen, C. (2019). An efficient collision power attack on AES encryption in edge computing. *IEEE Access, 7*, 18734–18748. https://doi.org/10.1109/ACCESS.2019.2896256
2. Bhanot, R., & Hans, R. (2015). A review and comparative analysis of various encryption algorithms. *International Journal of Security and its Applications, 9*(4), 299–306. https://doi.org/10.14257/IJSIA.2015.9.4.27
3. Guo, S., Zhao, X., Zhang, F., Wang, T., Shi, Z., Standaert, F. X., & Ma, C. (2014). Exploiting the incomplete diffusion feature: A specialized analytical side-channel attack against the AES and its application to microcontroller implementations. *IEEE Transaction on Information Forensics and Security, 9*(6), 999–1014.
4. Schneier, B. (2009). *Schneier on security: Another new AES attack*. Retrieved 5 Sept 2016 from https://www.schneier.com/blog/archives/2009/07/another_new_aes.html
5. CISCO. (2019). *Securing the internet of things: A proposed framework*. Retrieved 16 Jan 2019 from http://www.cisco.com/c/en/us/about/security-center/secure-iot-proposed-framework.html
6. Tentu, A. N. (2020). A review on evolution of symmetric key block ciphers and their applications. *IETE Journal of Education, 61*(1), 34–46. https://doi.org/10.1080/09747338.2020.1769508
7. Abd Elminaam, D. S., Kader, H. M. A., & Hadhoud, M. M. (2010). Evaluating the performance of symmetric encryption algorithms. *International Journal of Network Security, 10*(3), 213–219.
8. Thakur, J., & Kumar, N. (2011). DES, AES and blowfish: symmetric key cryptography algorithms simulation based performance analysis. *International Journal of Emerging Technology and Advanced Engineering, 1*(2), 6–12.
9. Mandal, P. C. (2012). Superiority of blowfish algorithm. *International Journal of Advanced Research in Computer Science and Software Engineering, 2*(9), 196–201.
10. Prakash, A., Lam, S. K., Clarke, C. T., & Srikanthan, T. (2013). FPGA-aware techniques for rapid generation of profitable custom instructions. *Microprocessors and Microsystems, 37*(2013), 259–269. https://doi.org/10.1016/j.micpro.2013.02.002
11. Devi, A., Sharma, A., & Rangra, A. (2015). Performance analysis of symmetric key algorithms: DES, AES, and blowfish for image encryption and decryption. *International Journal of Engineering and Computer Science, 4*(6), 12646–12651.
12. Patil, P., Narayankar, P., Narayan, D.G., & Meena, S.M. (2016). A comprehensive evaluation of cryptographic algorithms: DES, 3DES, AES, RSA and Blowfish. *In: Proceedings of the 2016 International Conference on Information Security & Privacy (ICISP2015)*, Procedia Computer Science, Elsevier, 78; pp. 617–624, Nagpur, India.
13. Rajasekaravarma, S., & Joshna, S. (2016). Symmetric key algorithms: a comparative analysis. *International Journal of Innovative Research in Computer and Communication Engineering, 4*(9), 15772–15775.
14. Silva, N. B. F., Pigatto, D. F., Martins, P. S., & Branco, K. R. L. J. C. (2016). Case studies of performance evaluation of cryptographic algorithms for an embedded system and a general purpose computer. *Journal of Network and Computer Applications, 20*, 13–143. https://doi.org/10.1016/j.jnca.2015.10.007
15. Sohal, M., & Sharma, S. (2018). BDNA-A DNA inspired symmetric key cryptographic technique to secure cloud computing. *Journal of King Saud University*, Computer and Information Sciences, 1–9, in press.
16. Nazeh Abdul Wahid, M. N., Ali, A., Esparham, B., & Marwan, M. D. (2018). A comparison of cryptographic algorithms: DES, 3DES, AES, RSA and blowfish for guessing attacks prevention. *Journal of Computer Science Applications and Information Technology, 3*(2), 1–7.

17. Schneier, B., & Whiting, D. (1997). Fast software encryption: designing encryption for optimal speed on the Intel Pentium processor. *In: Proceedings of 4th international workshop on fast software encryption*. LNCS, pp. 242–259. Springer Verlag.
18. Ghosh, S., & Karar, V. (2018). Blowfish hybridized weighted attribute-based encryption for secure and efficient data collaboration in cloud computing. *Applied Sciences, 8*(1119), 1–15. https://doi.org/10.3390/app8071119
19. Fan, C. P., & Hwang, J. K. (2008). FPGA implementations of high throughput sequential and fully pipelined AES algorithm. *International Journal of Electrical Engineering, 15*(6), 447–455.
20. Prasanthi, O., & Reddy, M. S. (2012). Enhanced AES algorithm. *International Journal of Computer Applications in Engineering Sciences, 2*, 114–118.
21. Subhashini, U., & Jagadeeswari, M. (2016). FPGA based encryption algorithm for secure communication. *International Journal of Innovative Research in Science, Engineering and Technology, 5*, 8707–8714.
22. Toubal, A., Bengherbia, B., Zmirli, M. O., & Guessoum, A. (2020). FPGA implementation of a wireless sensor node with built-in security coprocessors for secured key exchange and data transfer. *Measurement, 153*, 1–16. https://doi.org/10.1016/j.measurement.2019.107429
23. Elbirt, A.J., Yip, W., Chetwynd, B., & Paar, C. (2000). An FPGA implementation and performance evaluation of the AES block cipher candidate algorithm finalists. *In: Proceedings of the third advanced encryption standard candidate conference, national institute of standards and technology (NIST)*, pp. 13–27, New York.
24. Yoo, S. M., Kotturi, D., Pan, D. W., & Blizzard, J. (2005). An AES crypto chip using a high-speed parallel pipelined architecture. *Microprocessors and Microsystems, 29*, 317–326. https://doi.org/10.1016/j.micpro.2004.12.001
25. Adib, S. E., & Raissouni, N. (2012). AES encryption algorithm hardware implementation: throughput and area comparison of 128, 192 and 256-bits key. *International Journal of Reconfigurable and Embedded Systems, 1*(2), 67–74.
26. Wang, Y., & Ha, Y. (2013). FPGA-based 40.9-Gbits/s masked AES with area optimization for storage area network. *IEEE Transactions on Circuits and Systems-II: Express Briefs, 60*(1), 36–40. https://doi.org/10.1109/TCSII.2012.2234891
27. Neenu, S., & Bonifus, P. L. (2016). Design of AES architecture with area and speed tradeoff. *Elsevier Procedia Technology, 24*, 1135–1140.
28. Nuray, A., Beuchat, J. L., Okamoto, E., San, I., & Yamazaki, T. (2017). A low-area unified hardware architecture for the AES and the cryptographic hash function Grostl. *Journal of Parallel and Distributed Computing, 106*, 106–120. https://doi.org/10.1016/j.jpdc.2017.01.029
29. Guruprasad, S.P., & Chandrasekar, B.S. (2018). An evaluation framework for security algorithms performance realization on FPGA. *In: Proceedings of the 2018 IEEE international conference on current trends in advanced computing*, pp. 1–6, Bangalore, India.
30. Nabil, M., Khalaf, A. A. M., & Hassan, S. M. (2020). Design and implementation of pipelined and parallel AES encryption systems using FPGA. *Indonesian Journal of Electrical Engineering and Computer Science, 20*(1), 287–299. https://doi.org/10.11591/ijeecs.v20.i1.pp287-299
31. Kumar, T. M., Reddy, K. S., Rinaldi, S., Parameshachari, B. D., & Arunachalam, K. (2021). A low area high speed FPGA implementation of AES architecture for cryptography application. *Electronics, 10*(2023), 1–22. https://doi.org/10.3390/electronics10162023
32. Sudarshan, T.S.B., Mir, R.A., & Vijayalakshmi, S. (2005). DRIL-A flexible architecture for Blowfish algorithm encryption using dynamic reconfiguration, replication, inner-loop pipelining, loop folding techniques. *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)*. LCNS, pp. 625–639. Springer Verlag.
33. Karthigaikumar, P., & Baskaran, K. (2010). Partially pipelined VLSI implementation of blowfish encryption/decryption algorithm. *International Journal of Image and Graphics, 10*(3), 327–341.
34. Kurniawan, N.P., Purwanto, Y., & Darlis, D. (2014). An implementation of data encryption for internet of things using blowfish algorithm on FPGA. *In: Proceedings of the 2nd international conference on information and communication technology*, IEEE, pp. 75–79, Bandung, Indonesia. https://doi.org/10.1109/ICoICT.2014.6914043
35. Oukili, S., & Bri, S. (2016). High throughput parallel implementation of blowfish algorithm. *International Journal of Applied Mathematics and Information Sciences, 10*(6), 2087–2092.
36. Suresh, M., & Neema, M. (2016). Hardware implementation of blowfish algorithm for the secure data transmission in internet of things. *Elsevier Procedia Technology, 25*, 248–255. https://doi.org/10.1016/j.protcy.2016.08.104
37. Bansal, V. P., & Jassal, P. S. (2016). Synthesis and analysis of 64-bit blowfish algorithm using VHDL. *International Journal of Engineering Sciences, 17*(1), 316–322.

38. Chatterjee, S.R., Majumder, S., & Pramanik, B. (2014). FPGA implementation of pipelined blowfish algorithm. *In: Proceedings of 5th international symposium electronic system design*, pp. 208–209. https://doi.org/10.21817/ijet/2017/v9i2/170902320.
39. Nalawade, S., & Gawali, D.H. (2017). Design and implementation of blowfish algorithm using reconfigurable platform. *In: Proceedings of the 2017 international conference on recent innovations in signal processing and embedded systems*, IEEE, pp. 479–484, Bhopal, India.
40. Russell. (2019). *Nandland: What is setup and hold time in an FPGA?* Retrieved 25 May 2019 from https://www.nandland.com/articles/setup-and-hold-time-in-an-fpga.html
41. Xilinx, Inc. (2015). *Zynq-7000 all programmable SoC overview*. Product specification, DS190, v(1.8), USA.
42. Ahmad, R., Kho, D., Abd Manaf, A., & Ismail, W. (2019). Parallel-pipelined-memory-based blowfish design with reduced FPGA utilization for secure ZigBee real-time transmission. *Wireless Personal Communications, 104*, 471–489. https://doi.org/10.1007/s11277-018-6031-8
43. Xilinx, Inc. (2017). *User guide: Power analysis and optimization*, UG907, v2017.3, USA.

Rafidah Ahmad has received the B.Eng. and M.Sc. in Electrical and Electronic Engineering from Universiti Sains Malaysia (USM), Penang, Malaysia, in 2001 and 2005, respectively. She has now completed her Ph.D. in Electrical and Electronic Engineering specializing in wireless and mobile system at USM. Currently, she is a Senior Research Officer with Collaborative Microelectronic Design Excellence Centre (CEDEC), USM since 2005. Her research interest includes the development of cryptography for wireless communication and digital signal processing with ASIC and FPGA. To date she has produces more than 20 international publications as the first author including book chapter, journal papers, conference papers and lecture notes. She is also a senior member of IEEE.



Mohamad Faiz Mohamed Omar obtained his B.Eng. (Electronic Engineering) from Universiti Sains Malaysia, Malaysia in 2014, Master in RF and Microwave from Universiti Sains Malaysia (USM), Malaysia in 2017. Currently, he is a Research Officer at Collaborative Microelectronic Design Excellence Centre (CEDEC), Universiti Sains Malaysia, Penang, Malaysia. His current research interests include wireless circuit design, 5G cellular communication front-end and microwave link.

Jagadheswaran Rajendran is currently serving as a Senior lecturer at Collaborative Microelectronic Design Excellence Centre (CEDEC) and School of Electrical and Electronic Engineering, Universiti Sains Malaysia, working on CMOS analog IC Design, CMOS Radio Frequency (RF) IC Design and Monolithic Microwave Integrated Circuit (MMIC) Design. He received his B.Eng (Hons) from Universiti Sains Malaysia in 2004, M.Eng (Telecommunication) from Multimedia University in 2011 and Ph.D in the field of RFIC design from University of Malaya in 2015. He was with Laird Technologies as an Antenna Designer followed by serving Motorola Technology from 2005 to 2007 as R&D Engineer, working on mobile phone receiver system. In 2008, he joined BroadComm as MMIC designer, working mainly on GaAa based power amplifier, LNA and gain blocks, where he was elevated to the rank of Principal Engineer later. In 2015, he joined Silterra Malaysia, working on CMOS RFIC Design and device modelling. Till date, he has published more than 40 research papers, mainly journals and holds one US patent. He was the recipient of the IEEE Circuit and System Outstanding Doctoral Dissertation Award in 2015. He served as the Chairman of IEEE ED /MTT/SSC Penang Chapter in year 2011 and 2018. He is also a senior member of IEEE.

Widad Ismail is a Professor and the Project Coordinator for the Auto-ID Laboratory (AIDL), Universiti Sains Malaysia (USM), Penang, Malaysia. She received her B.Eng (H) First Class Honors in Electronics and Communication Engineering from The University of Huddersfield, United Kingdom in 1999. By 2004, she completed her Ph.D. in Electronics and Communication Engineering specializing in Active Integrated Antenna (AIA) with Image Rejection from the University of Birmingham, United Kingdom. Since year 2000, she served as a Postgraduate Teaching Assistant at the university for three years. Once graduated, she started her career at USM as a lecturer until today. She was appointed as professor in the year 2014 at the School Electrical & Electronics Engineering, USM. Her main areas of research are wireless sensor and system design, RFID (Radio Frequency Identification), active integrated antennas (AIA) and RF and microwave systems engineering. Her research and scientific outputs have been translated to numbers of awards, publications and patents. To date, she is a Principal Investigator for 26 research grants. These research works have produced 8 filed patents, 10 international awards, 4 commercialized main research products and more than 150 publications including the international journal papers, conference/seminars and other publications. Furthermore, several incomes are received to the University mainly from the Commercialization of research innovative products and also the services as a principle consultant. In addition, there are more than 35 consultations and collaborations that have been established with various agencies and institutions which bridging the gap between the academicians to the industrialists. Currently, she is the main supervisor of 13 PhD students (active candidature) and 4 Master's by research students and she has graduated a total of 12 PhD and 12 Master's students under her supervision and guidance. On top of these, she is a member of Wireless World Research Forum (WWRF).

Authors and Affiliations

Rafidah Ahmad[1] · Mohamad Faiz Mohamed Omar[1] · Jagadheswaran Rajendran[1] · Widad Ismail[2]

✉ Rafidah Ahmad
rafidah.ahmad@usm.my

Mohamad Faiz Mohamed Omar
faiz_omar@usm.my

Jagadheswaran Rajendran
jaga.rajendran@usm.my

Widad Ismail
eewidad@usm.my

[1] Collaborative Microelectronic Design Excellence Centre (CEDEC), Engineering Campus, Universiti Sains Malaysia, 14300 Nibong Tebal, Pulau Pinang, Malaysia

[2] Auto-ID Laboratory, School of Electrical & Electronic Engineering, Engineering Campus, Universiti Sains Malaysia, 14300 Nibong Tebal, Pulau Pinang, Malaysia