applied sciences

IMPACT FACTOR 2.838

CITESCORE 3.7

Article

Maintaining Secure Level on Symmetric Encryption under Quantum Attack

Hung-Jr Shiu, Chao-Tung Yang, Yun-Ru Tsai, Wei-Chung Lin and Chun-Ming Lai

MDPI

https://doi.org/10.3390/app13116734

applied sciences

MDPI

Article

Maintaining Secure Level on Symmetric Encryption under Quantum Attack

Hung-Jr Shiu [1], Chao-Tung Yang [2,3], Yun-Ru Tsai [2], Wei-Chung Lin [1] and Chun-Ming Lai [2,*]

[1] Department of Computer Science and Information Engineering, National Taipei University, New Taipei City 237, Taiwan
[2] Department of Computer Science, Tunghai University, Taichung City 407, Taiwan
[3] Research Center for Smart Sustainable Circular Economy, Tunghai University, Taichung City 407, Taiwan
* Correspondence: cmlai@thu.edu.tw

Abstract: Quantum computing is currently being researched in many countries, and if implemented in the near future, it may pose a threat to existing encryption standards. In the quantum computer environment, asymmetric encryption can be solved by Shor's Algorithm in polynomial time, and the difficulty of breaking symmetric encryption using brute force is reduced from N times to square root N times by Grover's Algorithm. We take the Advanced Encryption Standard as the theme and increase the key length from the original standard 192 bits and 256 bits to 384 bits and 512 bits, respectively, in order to maintain the security level of AES 192/256 under the environment of quantum computing, so we propose the key schedule of AES 384/512, and write the software in C++ on FPGA. The experimental results show that our scheme can achieve Level III and Level V security levels in a quantum computer attack environment. In addition to increasing the length of the key, we use the LUT method in the process of writing SubBytes to replace the array and speed up the computation to optimize the execution speed. In addition, the proposed scheme is still based on 128-bit computing blocks, rather than computing blocks in larger blocks.

Keywords: quantum computing; Shor's algorithm; Grover's algorithm; advanced encryption standard; strict avalanche criterion; Rijndael algorithm

Citation: Shiu, H.-J.; Yang, C.-T.; Tsai, Y.-R.; Lin, W.-C.; Lai, C.-M. Maintaining Secure Level on Symmetric Encryption under Quantum Attack. *Appl. Sci.* **2023**, *13*, 6734. https://doi.org/10.3390/app13116734

Academic Editor: Luis Javier García Villalba

Received: 25 April 2023
Revised: 23 May 2023
Accepted: 24 May 2023
Published: 31 May 2023

1. Introduction

Recent years have seen the continued progress of data communication and application. Therefore, security systems [1] and equipment to protect personal information [2,3] and its transmission channels are indispensable. One of the most important technologies is data encryption, which is mainly used to protect information.

Symmetric and asymmetric cryptosystems are the most widely used in data encryption and symmetric encryption techniques such as the Data Encryption Standard (DES), Triple DES, and the Advanced Encryption Standard (AES) [4]. The sender and receiver use the same key to achieve data encryption and decryption. Among them, the AES published by the National Institute of Standards and Technology (NIST) in 2001 is the most outstanding among symmetric encryption technologies. AES uses SubBytes, ShiftRows, MixColumns, and AddRoundKey, which are the four main transformation equations, to form the main structure.

However, with the development of science and technology, at the same time, the technology of quantum computing is also constantly improving. The asymmetric encryption algorithm can be solved by Shor's algorithm [5] in polynomial time in a quantum computer environment, and its security is equivalent to direct disintegration. The difficulty of the brute force cracking of the symmetric encryption algorithm will be diminished by Grover's Algorithm [6] from N times to square root N times, and its security will also be reduced a lot. Although both algorithms will be affected by quantum computers, the AES, which is a member of symmetric encryption, will be less affected. Only the security level of the

original key length will be cut in half; that is, the highest security level of AES 256 will become as low as AES 128.

In order to ensure that the AES [7] maintains the original security level in the quantum computer environment, we decided to expand according to Rijndael's theory and try to increase the encryption key to maintain the original security strength. We use Field Programmable Gate Array (FPGA) to achieve verification, and the overall algorithm acceleration and optimization to achieve the effect of hardware accelerated computing.

We propose the AES algorithm for countering quantum computing to improve the security of the algorithm. The following are the main contributions of this paper:

- To follow the framework of the original AES algorithm, and extend the key length to 384 bits and 512 bits in order to resist the threat introduced by quantum computers, and do an avalanche test on C++ software to prove the security of the AES 384 bits and 512 bits key.
- Improve the speed of AES encryption and decryption, write the AES algorithm into FPGA by using hardware description language, make use of its instruction parallelization characteristics, make parallel calculations, accelerate the encryption and decryption calculation, and achieve a hardware acceleration effect.
- Use different calculation structures and increase the amount of calculation per unit of time by adding registers, and use the space-for-time method for optimization.

In Section 2, a literature review on the state of quantum computing, AES, etc. will be undertaken. In Section 3, the basic architecture and calculation method proposed in this paper will be described. In Section 4, it is described how to implement the proposed method on FPGA, which involves implementation performance comparison and simulation implementation screen. Finally, the conclusion of this work is stated in Section 5.

2. Literature Review

2.1. Quantum Computer

Quantum computing is a method that uses a quantum as the basic computing [8] unit and operates with quantum algorithms. As long as a quantum reaches the quantum superposition state and the quantum entanglement state at the same time, it becomes a qubit, providing a calculation. The reading and writing of qubits can be done using laser or microwave techniques. The current mainstream quantum computing methods are divided into five types: silicon-based spin qubits, ion traps, diamond nitrogen vacancy centers, topological qubits, and superconducting rings. In 2021, IBM released Eagle with 127 bits, and it is expected for this to increase to 433 bits in 2022.

The difference between a quantum computer and a conventional computer is that quantum computers exploit the properties of quantum entanglement and superposition. Because of this unique property, computers made of qubits can process large amounts of data in parallel.

However, the computing power of current quantum computers is not sufficiently strong [9,10], therefore there are still has many problems to overcome.

2.2. Advanced Encryption Standard

This algorithm is a NIST-approved symmetric encryption block encryption method, also known as the Rijndael algorithm. In 1997, NIST announced that it would choose a better algorithm to replace the DES algorithm that was widely used at that time. The biggest disadvantage of DES is that the key is only 56 bits, which is relatively small, and it is more and more vulnerable to attack. In 2001, the Rijndael algorithm was selected as the AES [4], because the Rijndael encryption method can support a wider range of block and key lengths.

AES is encrypted and decrypted in units of blocks, each block is fixed at 128 bits, and the key has three lengths of 128 bits, 192 bits, and 256 bits to choose from. According to the key length, the encryption rounds are also divided into 10, 12, and 14 rounds. The

AES encryption workflow of each round includes the following four steps: AddRoundKey, SubBytes, ShiftRows, and MixColumns.

AddRoundKey: Each byte in the matrix is XORed with the round key, and each sub-key is generated by the key generation scheme.

SubBytes: Each byte is replaced with the corresponding byte using a lookup table through a non-linear replacement function.

ShiftRows: A circular shift is applied to each row in the matrix.

MixColumns: In order to fully mix the operations of each column in the matrix, this step uses a linear transformation to mix the four bytes within each row. The MixColumns step in the last encryption loop is omitted and replaced with another AddRoundKey.

After 10 to 14 rounds of repeated calculations, the complexity of the brute force cracking method can reach up to 2^{256} times, which ensures the security strength of the AES on traditional computers.

2.3. Finite Field

Also called the Galois field, it is based on the Galois finite field theorem [11]; if it a positive prime number and a positive integer, then there is a finite field containing elements, and the structure of the finite fields with elements is absolutely isomorphic.

Finite fields are widely used in modern coding, theoretical computer science, combinatorics, and cryptography. The AES algorithm proposed in this paper uses the Galois field for calculation, and the finite field calculation is divided into two types.

The first type is finite field addition, where adding a finite field is a simplification of a polynomial modulo, and its eigenvalues ensure that the result of the computation stays in the finite field. AES uses the Galois field, and its eigenvalue is 2, so the addition needs to be modulo 2, which can be simplified to the XOR equivalent to the logical operation. The second type is finite field multiplication, and like finite field addition, when the result of multiplying two polynomials may be larger than nth power, the value will exceed the range of finite field, so the finite field multiplication also needs to go through the modulo operations. The modulus used by finite field multiplication must be an integrable polynomial, while the AES algorithm uses AND, which is equivalent to the logical operation.

2.4. Avalanche Effect

This term was first used by Horst Feistel [12], and the origin of the concept can be traced back to the dissemination by Claude Shannon. The term diffusion refers to the redundancy of plain text in cryptography to spread the effects of a single key to as many ciphertexts as possible and avoid the possibility of deciphering the input from the output using brute force methods.

The avalanche effect is designed to do the same, and is considered an important metric in cryptographic security to ensure that text or keys cannot be corrupted by statistical analysis.

The Strict Avalanche Criterion [13] was proposed by Webster and Tavares as a formalization of the avalanche effect. If the encryption method meets the strict avalanche criteria, when any input bit is inverted, there is a 50% chance that each bit of the output will change.

2.5. Parallel Computing

Parallel computing refers to the use of multiple computing resources to improve the efficiency of data processing, and parallel computing can be broadly divided into spatial and temporal parallelism. Time parallelism refers to pipelining, which is splitting instructions into multiple steps so that they can be processed in parallel to speed up instruction execution; spatial parallelism involves the use of multiple computing resources at the same time to improve computation speed. In this case, we programmed the AES C++ code into a hardware description language [14] in a pipelined manner so that it can be executed on FPGAs to achieve hardware acceleration.

2.6. Field Programmable Gate Array (FPGA)

A Field Programmable Gate Array (FPGA) [14] is a logic circuit written in a hardware description language and is an actual parallel architecture. Generally speaking, FPGA is slower than ASIC and is not suitable for designing very complex circuit structures. At the same time, the power consumed by FPGA is larger than that of ASIC. However, the biggest advantage of FPGAs is that they can be manufactured quickly, and the internal logic can be modified repeatedly, making debugging costs lower. In addition, FPGAs can perform a task at exactly the same speed over and over again, making them suitable for developing low-latency specialized chips.

2.7. Abidalrahman Moh'd Proposed AES-512 Bits Method

Abidalrahman Moh'd et al. [15] proposed the AES enhanced algorithm for quantum computing, and the length of the original encrypted key was increased from 256 bits to 512 bits. In order to be able to decrypt it, the original length of 128 bits was changed to 512 bits on the FPGA, but there are several disadvantages involved with this method.

First, unlike the original AES architecture, the security cannot be compared. Second, since each block expands in length, the overall space complexity is greatly increased. Third, since the block length is at least 512 bits, security from Level-1 to Level-3 cannot be achieved.

3. System Architecture

3.1. AES Algorithm Parameters

The AES algorithm has the following three parameters:

1. The number of encrypted blocks (*Nb*): The input plaintext, in 32 bits, is 1 word, because the AES plaintext block is fixed at 128 bits, which is the state matrix composed of four words.
2. Number of key segments (*Nk*): The length of the key is 32 bits for one word, and is divided into four words, six words, and eight words according to the key length of 128 bits, 192 bits, and 256 bits.
3. Number of encryption rounds (*Nr*): This is the number of iterations required for encryption and decryption, which is related to the key length. The number of rounds from short to long key length is 10, 12 and 14 rounds, respectively. The relationship between the number of encryption rounds and the number of key segments can be calculated as follows:

$$Nr = 6 + Nk \quad (1)$$

The above parameters, according to the different key lengths, can be organized as shown in Table 1:

Table 1. Important AES parameters.

Key Length (Bits)	Number of Encrypted Segments (*Nb*)	Number of Key Segments (*Nk*)	Number of Encrypted Rounds (*Nr*)
128	4	4	10
192	4	6	12
256	4	8	14

3.2. AES-384, AES-512 Concept and Software Implementation

In order to maintain the security level of AES in a quantum computer environment, it is necessary to extend the length of the original key, and this paper proposes two versions of AES-384 and AES-512 with an extended key. The proposed new schedule must conform to the security framework of the Rijndael algorithm, so we extend it based on the original framework [16]. The differences between the two new key schedules and the original architecture are only the number of cryptographic rounds and the key expansion steps, except for the key length.

Both AES-384 and AES-512 execute only AddRoundKey in the first round, and execute SubBytes, ShiftRows, MixColumns and AddroundKey in the second round to the $Nr - 1$ round In the final round, it executes SubBytes, ShiftRows and AddRoundKey. We then calculate the number of needed rounds by the formula: $Nr = 6 +$ bits of key$/32$. The results are shown in Table 2.

Table 2. Number of rounds for AES 384/512.

Version	Round
384	18
512	22

Turning now to the key expansion, the first step is calculating the needed words by using the formula $i = Nb * (Nr + 1)$. Note whether the number of subkeys is correct. The results of the calculations are presented in Table 3 sketch pictures of key expansions according to calculations. The figures make it easy to understand the work. Figures 1 and 2 illustrate the process of key expansion for AES-384 and AES-512.

Table 3. Number of keys and words for AES 384/512.

Version	*Nr*	Words
384	18	76
512	22	92

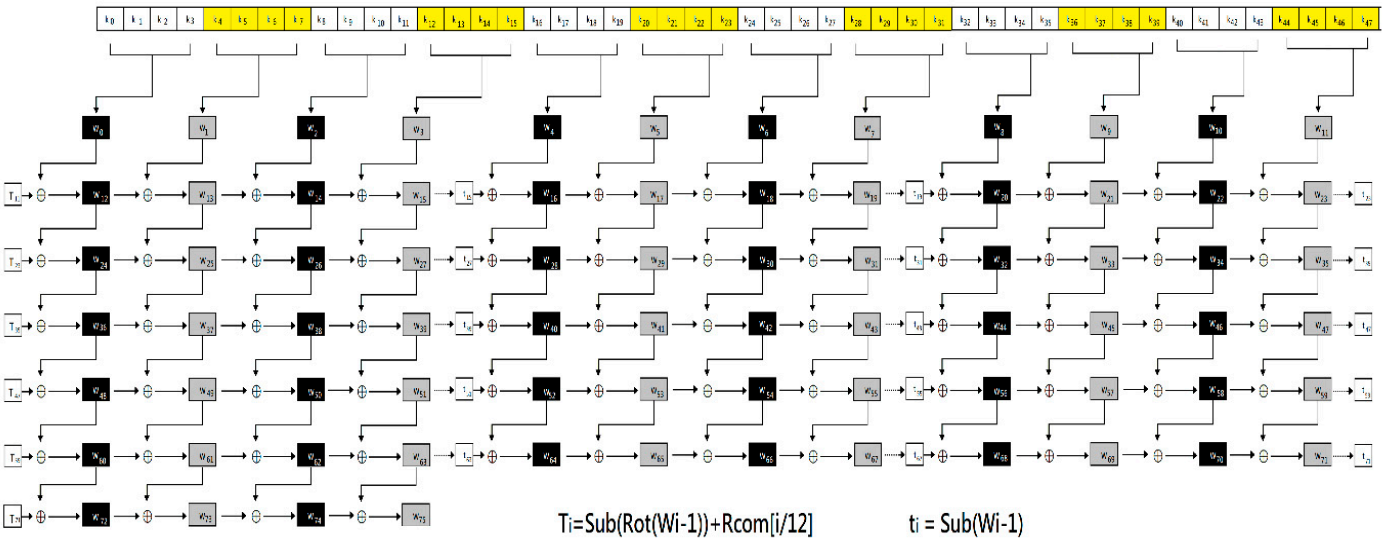

Figure 1. AES 384 key expansion.

From Tables 4 and 5, we know that AES-384/512 is extended by the original architecture, so the original encryption standard will not be changed, thus affecting security. The benefit of designing AES-384 and AES-512 by extending from the original standard is that the use of this method can reduce the risk posed to AES-192 and AES-256.

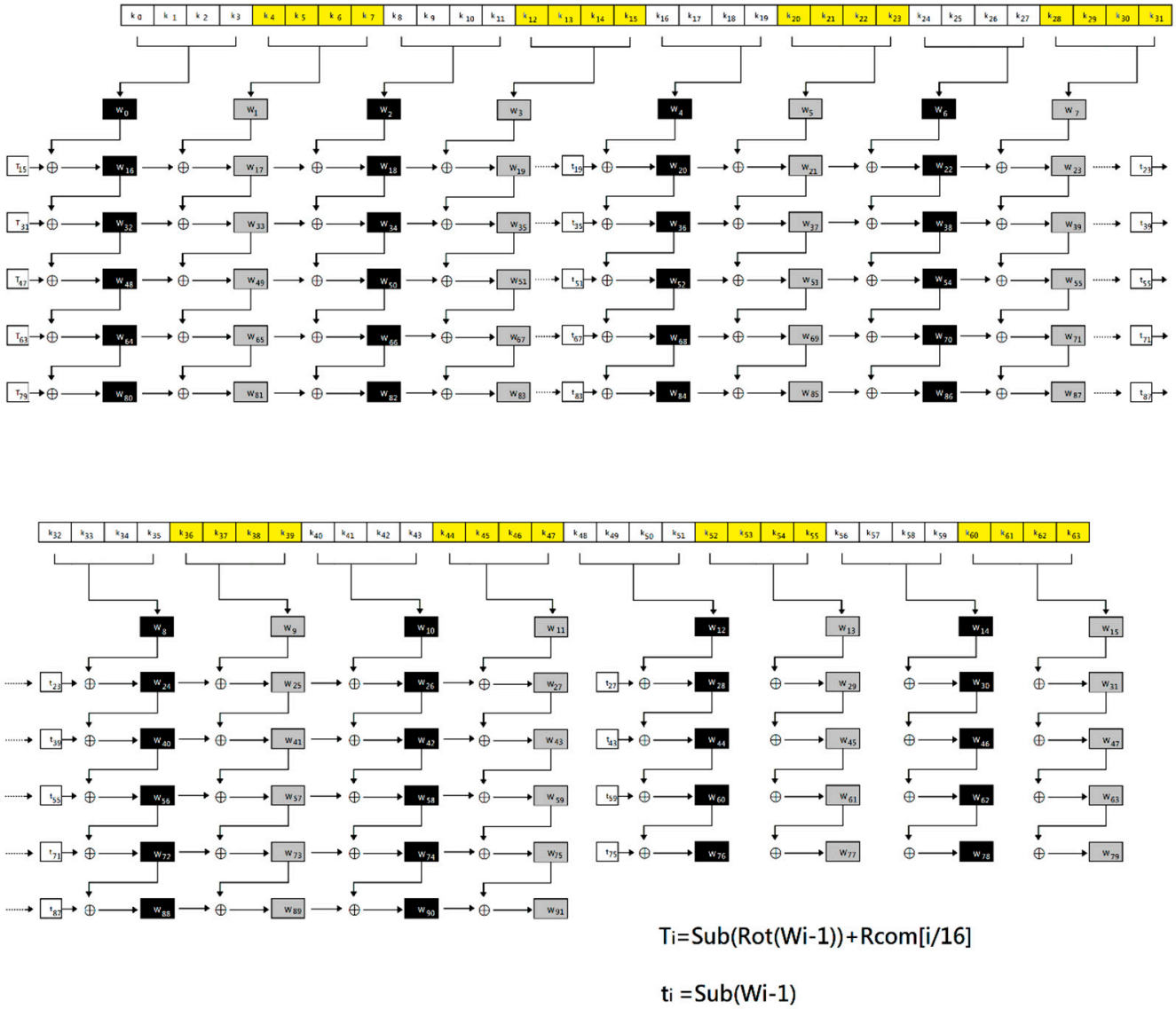

Figure 2. AES 512 key expansion.

Table 4. AES 384 key expansion rule.

Block Number	Rule
$i \bmod 12 = 0$	$Wi = Sub(Rot(Wi-1))$
$i \bmod 12 \neq 0$	$Wi = Wi-1+Wi-12$
$i \bmod 12 \neq 0$ & $i \bmod 8 \neq 0$ & $i \bmod 4 = 0$	$Wi = Sub(Wi-1)+Wi-12$
$i \bmod 12 \neq 0$ & $i \bmod 8 = 0$	$Wi = Sub(Wi-1)+Wi-12$

Table 5. Key expansion of AES 512.

Block Number	Rule
$i \bmod 16 = 0$	$Wi = Sub(Rot(Wi-1))+Rcom\left[\frac{i}{16}\right]+Wi-16$
$i \bmod 16 \neq 0$	$Wi = Wi-1+Wi-16$
$i \bmod 16 \neq 0$ & $i \bmod 12 \neq 0$ & $i \bmod 8 \neq 0$ & $i \bmod 4 = 0$	$Wi = Sub(Wi-1)+Wi-16$
$i \bmod 16 \neq 0$ & $i \bmod 12 \neq 0$ & $i \bmod 8 = 0$	$Wi = Sub(Wi-1)+Wi-16$
$i \bmod 16 \neq 0$ & $i \bmod 12 = 0$	$Wi = Sub(Wi-1)+Wi-16$

4. Programming and Testing

4.1. Experimental Environment

The development kit used was a Xilinx Vivado 2020.2 for coding and simulation, while the hardware system was a Zynq 7000 series: SoC XC7Z7100 FPGA development board. Table 6 summarizes the number of hardware objects used to implement AES-384/512.

Table 6. The number of hardware objects used to implement AES-384/512.

Items of Hardware Objects	AES-384 Bits	AES-512 Bits
Number of Slices	4879	6371
Number of Slices FF	2791	3791
Number of four-input LUTs	8997	11,211
Number of bonded IOBs	471	594

According to the original theory, in AES-384, the number of Slices should be approximately 4900 and 6500; the number of Slices FF should be about 2900 and 3800; the number of four-input LUTs should be about 9100 and 12,100; the number of bonded IOBs should be about 500 and 660, but because of the implementation, we will not need the unit or reuse the unit, so it will be less than that indicated by the theory.

4.2. Software Testing

This section shows the results of the software implementation using AES-128 [17]. This test uses the standard S-box, Inverse S-box, Mixcolumns and InvMixColumns from NIST AES, and the explicit text is "Learn to walk before you run". The following chart includes the complete contents of the S-box and Mixcolums, as well as the initial key and the result of each round key expansion.

Table 7 [7] is the 10-round expanded key generated by the secret key, and Table 8 [4] is the S-box of the AES software test.

Table 7. S-box matrix.

	00	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0a**	**0b**	**0c**	**0d**	**0e**	**0f**
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	Af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Table 8. 10 round expanded key.

Original Key	**2b**	**7e**	**15**	**16**	**28**	**ae**	**d2**	**a6**	**ab**	**f7**	**15**	**88**	**09**	**cf**	**4f**	**3c**
Round 1 key	a0	fa	fe	17	88	54	2c	b1	23	a3	39	39	2a	6c	76	05
Round 2 key	f2	c2	95	f2	7a	96	b9	43	59	35	80	7a	73	59	f6	7f
Round 3 key	3d	80	47	7d	47	16	fe	3e	1e	23	7e	44	6d	7a	88	3b
Round 4 key	ef	44	a5	41	a8	52	5b	7f	b6	71	25	3b	db	0b	ad	ee
Round 5 key	d4	d1	c6	f8	7c	83	9d	87	ca	f2	b8	bc	11	f9	15	bc
Round 6 key	6d	88	a3	7a	11	0b	3e	fd	db	f9	86	41	ca	00	93	fd
Round 7 key	4e	54	f7	Be	5f	5f	c9	f3	84	a6	4f	b2	4e	a6	dc	4f
Round 8 key	ea	d2	73	21	b5	8d	ba	d2	31	2b	f5	60	7f	8d	29	2f
Round 9 key	ac	77	66	f3	19	fa	dc	21	28	d1	29	41	57	5c	09	6e
Round 10 key	d0	14	f9	a8	c9	ee	25	89	e1	3f	0c	c8	b6	63	0c	a6

4.3. Software Avalanche Test

In order to ensure the security of AES-384 and AES-512, two scheduled avalanche tests will be conducted in this section. First, the former will be shown and explained, and the latter will only include the execution process [18].

The avalanche test of AES-384 uses two strings with a difference of only one character. Table 9 presents the input and output of two plaintexts.

Finally, Table 10 shows the results of two kinds of plaintext encryption, including hexadecimal and binary demonstrations. This test has a total of 128 bits, and the two strings of very similar plaintexts have changed by 64 bits, which is exactly 50%, which meets the conditions of the avalanche test.

Table 9. AES-384 input plaintext.

Text Characters	**Plaintext 1** **Helloworld123456**								**Plaintext 2** **Helloworld123450**							
ASCII	48	65	6c	6c	6f	77	6f	72	48	65	6c	6c	6f	77	6f	72
	6c	64	31	32	33	34	35	36	6c	64	31	32	33	34	35	30
Original key (384 bits)	2b	7e	15	16	28	ae			d2	a6	ab	f7	15	88		
	09	cf	4f	3c	7e	15			16	28	ae	d2	a6	ab		
	f7	15	88	09	cf	4f			3c	2b	2b	7e	15	16		
	28	ae	d2	a6	ab	f7			15	88	09	cf	4f	3c		

Table 10. AES-384 output plaintext.

	Plaintext 1								**Plaintext 2**							
Output	8c	09	93	51	a6	3e	92	8e	f2	99	04	96	69	31	36	b1
ASCII	d6	99	ba	2f	ae	53	6f	b6	55	98	13	b1	4a	77	33	9b

Next is the AES-512 avalanche test process and results, and the test data used in AES-512. Table 11 shows the input information of two different plaintext encryptions. Table 12 shows the two kinds of plaintext output results. According to the actual measurement of the encryption results, the two gaps between the latter accounted for 53.9% of the total number of bits which also passed the avalanche test.

Table 11. AES-512 input plaintext.

	Plaintext 1								Plaintext 2							
Text Characters	**Thatisagoodidea8**								**Thatisagoodidea5**							
ASCII	54	68	61	74	69	73	61	67	54	68	61	74	69	73	61	67
	6f	6f	64	69	64	65	61	38	6f	6f	64	69	64	65	61	35
Original key (384 bits)	2b	7e	15	16	28	ae	d2	a6	ab	f7	15	88				
	09	cf	4f	3c	7e	15	16	28	ae	d2	a6	ab				
	f7	15	88	09	cf	4f	3c	2b	2b	7e	15	16				
	28	ae	d2	a6	ab	f7	15	88	09	cf	4f	3c				
	7e	15	16	28	ae	d2	a6	ab	f7	15	88	09				
	2b	7e	15	16	28	ae	d2	a6	ab	f7	15	88				
	09	cf	4f	3c	7e	15	16	28	ae	d2	a6	ab				
	cf	4f	3c	2b												

Table 12. AES-512 output plaintext.

	Plaintext 1								Plaintext 2							
Output	2f	c4	ae	d4	f6	94	24	8f	c3	1a	83	13	c3	01	f1	fb
ASCII	a4	93	64	fb	f1	28	80	02	aa	7f	1c	46	9f	98	23	20

4.4. Hardware Description Language Simulation

This section will show the AES-384 and AES-512 hardware description language [19–22] simulation process, mainly AES-384/512 for explanation, only with AES-384/512 attached to each process of the demonstration diagram [23–25], and not attached versions of AES-128, AES-192, or AES-384. The test content is "Learn to walk before you run".

Figure 3 shows the complete information and results of the AES-384 encryption and decryption, including the parameters used in AES, standard S-box [26–29], inverse S-box, Mixcolumns and InvMixColumns, as well as the input plaintext, encrypted ciphertext, key expansion, and ETC.

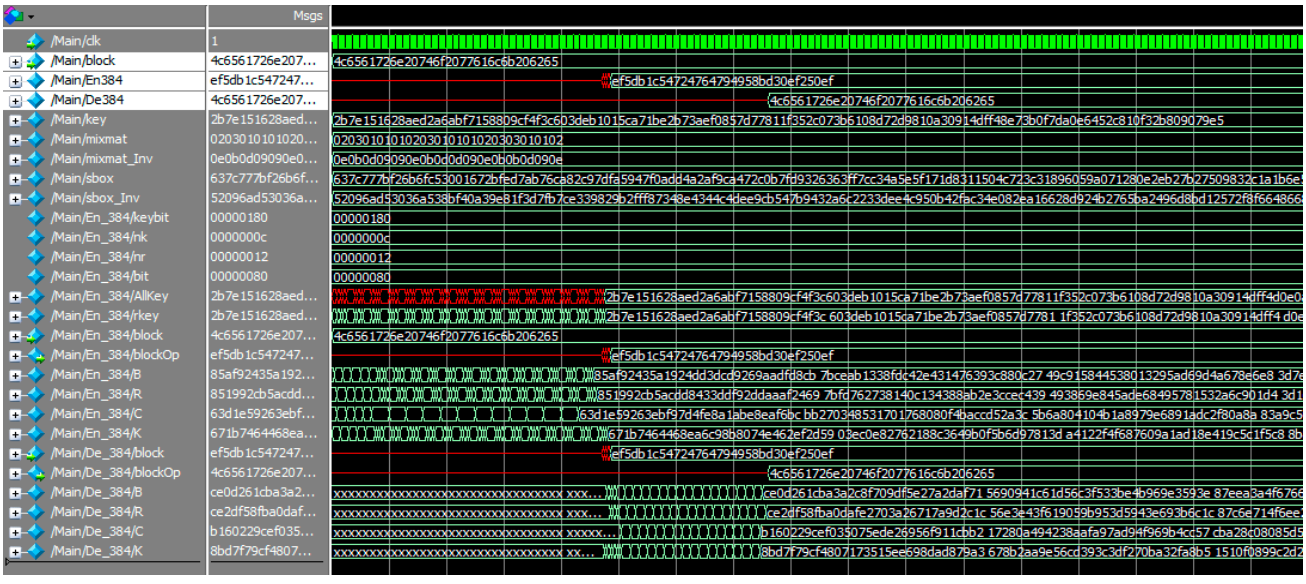

Figure 3. AES-384 encrypted and decrypted simulation.

Figure 4 shows the results of each round of AES-384 key expansion, and it can be seen that the value of each round key and the simulated time delay are step by step. Each round needs to wait for the execution result provided by the previous round, and it proceeds to the next round immediately after some results are outputted.

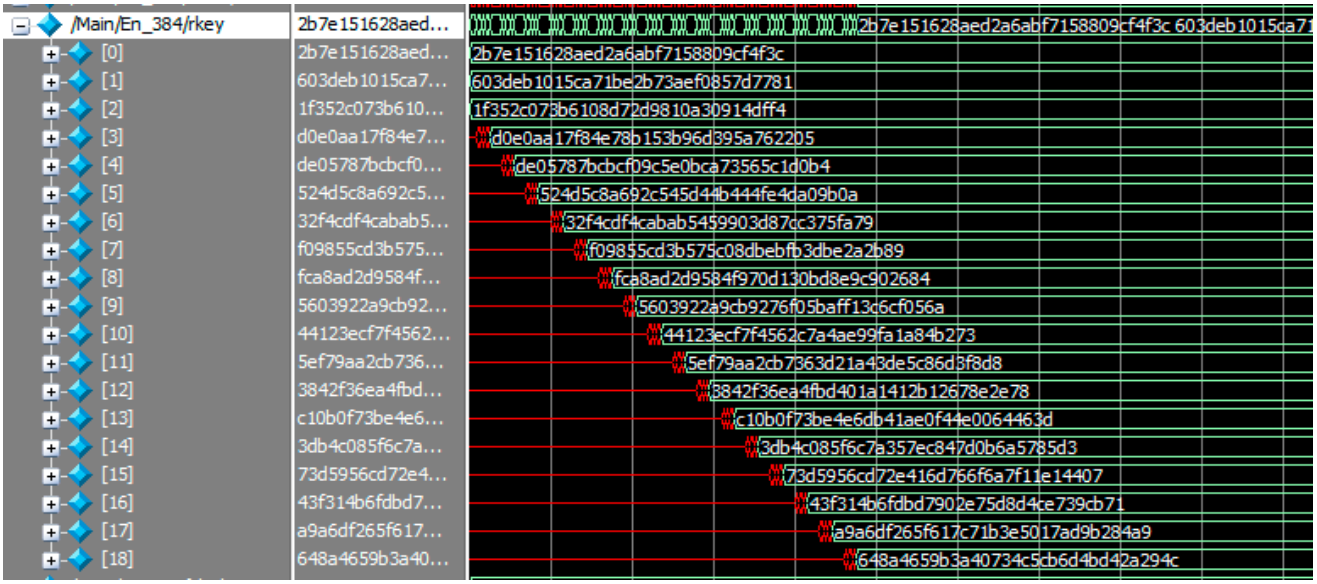

Figure 4. AES-384 key expansion simulation.

Figure 5 shows a series of AES-512 encryption and decryption simulations and results. The most obvious difference between AES-512 and AES-384 is that AES-512 performs more rounds, and Figure 6 shows the results of each round of the AES-512 key expansion.

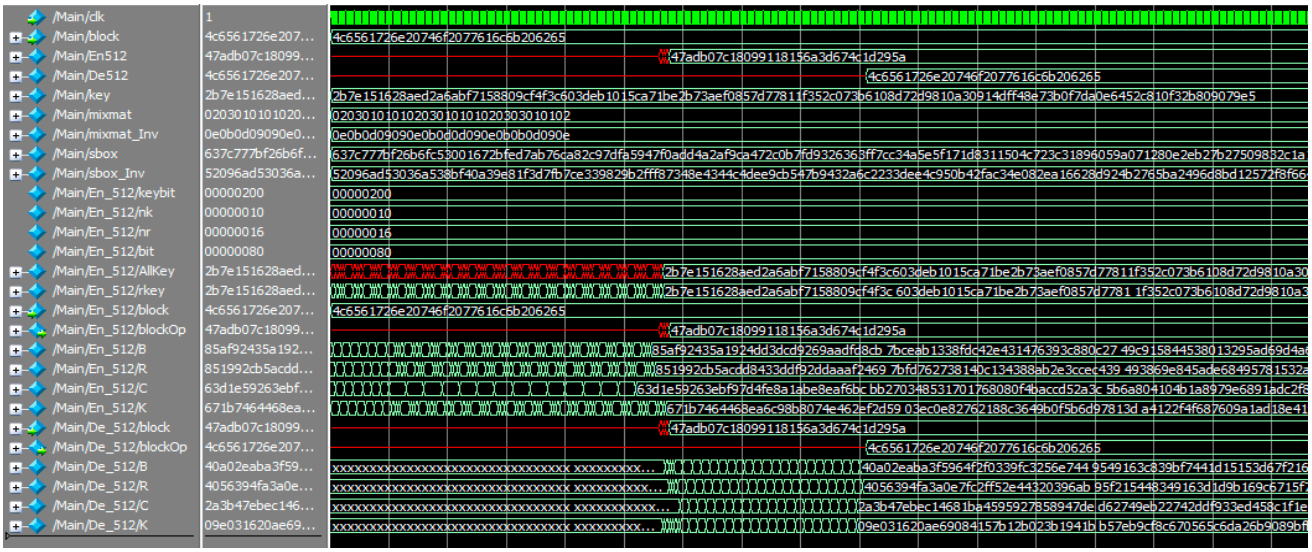

Figure 5. AES-512 encrypted and decrypted simulation.

Although this section does not attach the simulation process of AES-128, AES-192 and AES-256, Figure 7 illustrates the execution results of all versions from AES-128 to AES-512, which proves that we have integrated the five keys of the hardware description of AES (the language of the length version).

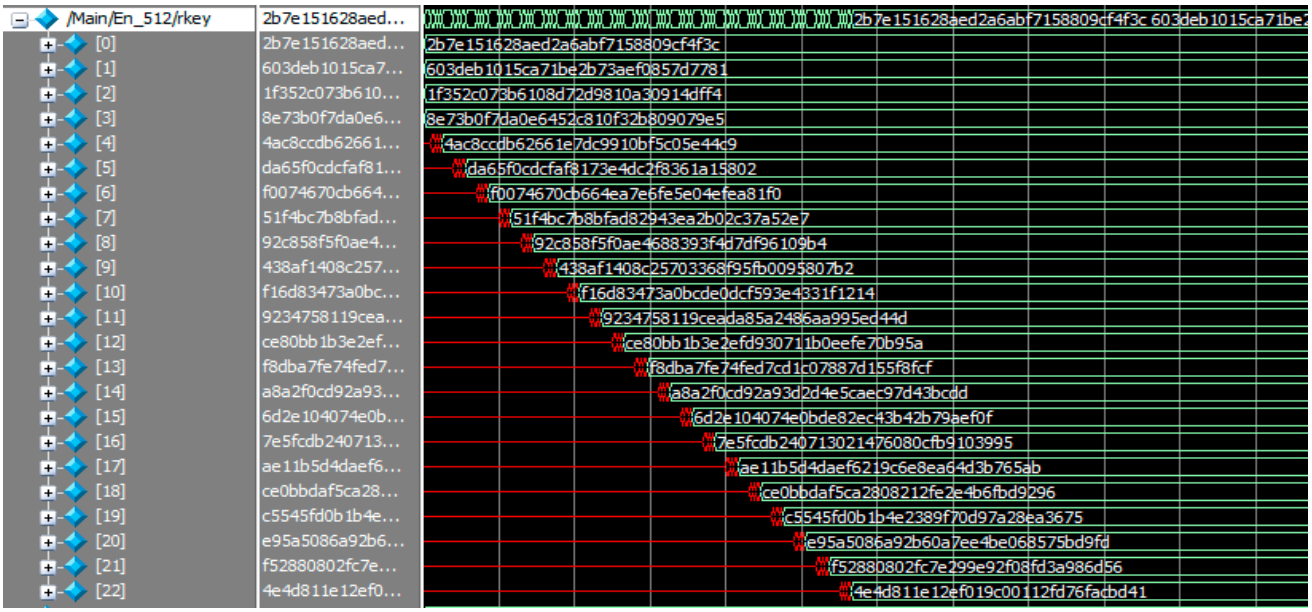

Figure 6. AES-512 key expansion simulation.

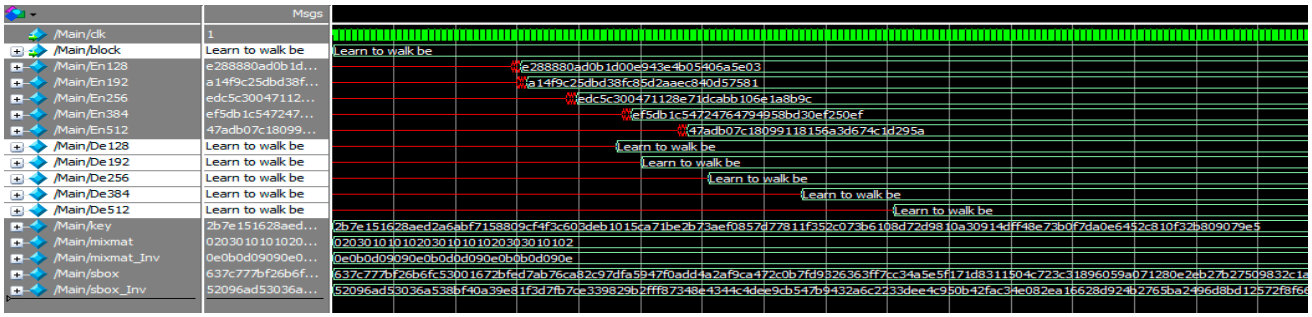

Figure 7. AES All versions simulation result.

4.5. Compatibility of Rijndael

Table 13 compares the proposed AES-384 and AES-512 with the original AES function of the paper [15].

Table 13. Compatibility comparison.

	Proposed	Rijndael	Abidalrahman Moh'd [15]
Security level	I, III, V	I, III, V	X
Security Level under quantum calculation	I, III, V	I	V
AES-128	O	O	X
AES-192	O	O	X
AES-256	O	O	X
AES-384	O	X	X
AES-512	O	X	O

The experimental results include three methods: the method proposed in this article, the method of Rijndael, and [15]. The proposed method and the Rijndael method can reach FIPS security levels I, III, and V under normal computer attacks, while [15] is a special architecture with a minimum key length of 512 bits, such that it cannot support any level other than level V. Under the FIPS security level of a quantum computer attack, the

proposed method can still maintain level I, III, and V, while Rijndael can only maintain level I, and [15] can only maintain level V.

Finally, the authors' scheme obtains the calculation function of AES-128/192/256/384/512 bits, Rijndael obtains the calculation function of AES-128/192/256 bits, and [15] only obtains the calculation function of AES-512 bits.

4.6. Parallel Computing Performance

Figure 8 demonstrates the time comparison of the general calculation of AES-384 and the parallel calculation of AES-384, while Figure 9 shows the time comparison of the general calculation of AES-512 and the parallel calculation of AES-512 under CTR mode.

From the above results, it can be seen that in the original normal calculation, when the file size is 1762 bits, the execution time of AES-384 is 18.74 ms, and the execution time of AES-512 is 23.14 ms. With the addition of parallel calculations, it can be accelerated to 0.008927 ms and 0.009378 ms; when the file size is 56384 bits, the execution time of AES-384 is 554.50 ms, and the execution time of AES-512 is 682.62 ms. After a parallel calculation, it can be completed within 0.008239 ms and 0.010082 ms. We know from the experimental results that we can speed up this process by parallelizing the overall computing speed.

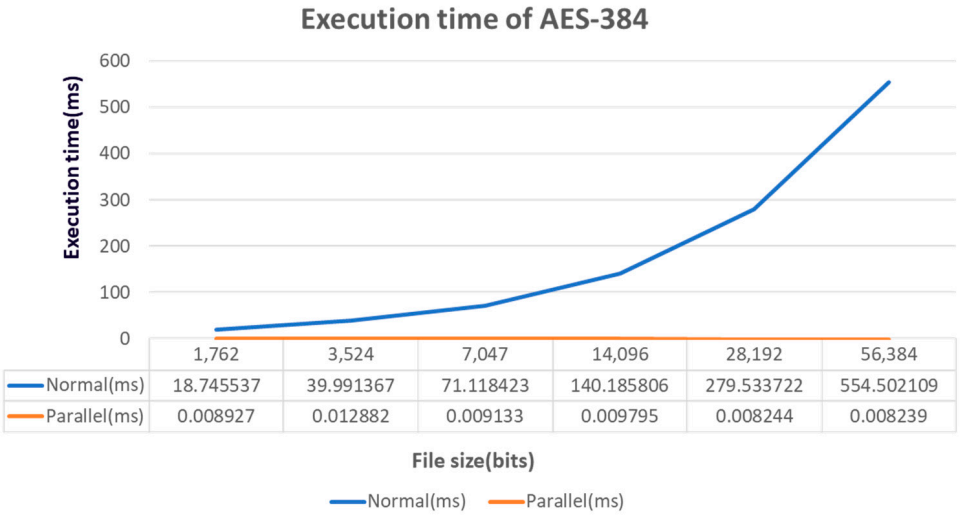

Figure 8. Execution time of normal AES-384 and parallel AES-384.

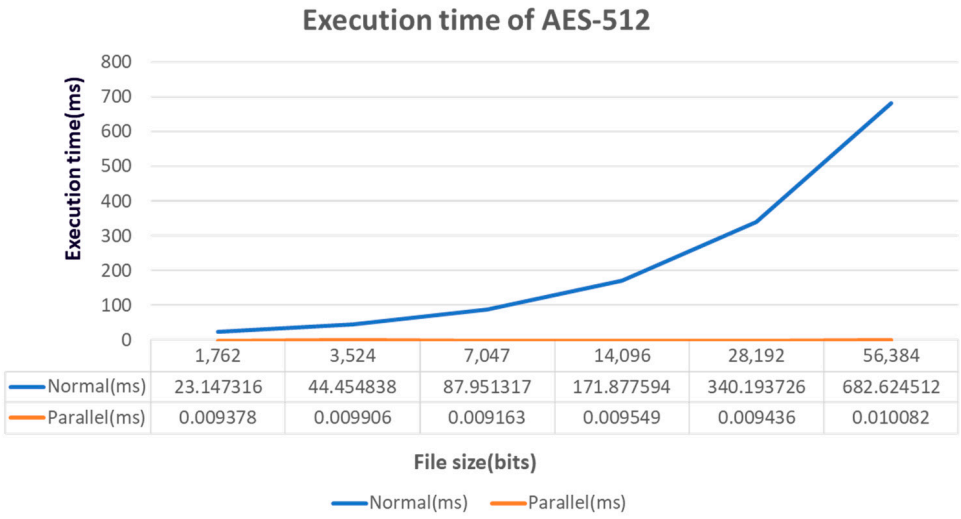

Figure 9. Execution time of normal AES-512 and parallel AES-512.

5. Conclusions

In order to make AES resistant to the threat of the quantum computer environment, this project implemented the key scheduling extension. In addition to software implementation, we hope to speed up the calculation speed of encryption and decryption. Therefore, we also implemented the hardware description language of AES-384 and AES-512 so that they can run smoothly on FPGA and achieve the effect of hardware acceleration.

In addition to basic hardware acceleration, we have also been working hard to optimize the hardware description language. By increasing the memory and cutting smaller input segments, parallel computing can be more complete, and the amount of computation per unit time can be increased.

Given the final experimental results shown in Table 12, our proposed method successfully integrates the original AES-128, AES-192, and AES-256 architectures with the newly arranged AES-384 and AES-512 in both hardware and software. This integration allows for the utilization of the complete set of advanced encryption algorithms. The standard still maintains the Level III and Level V levels in the secure quantum computer attack environment, and there is no need to change the original encryption architecture of Rijndael, as [10] did, which successfully achieves the purpose of this research project.

Author Contributions: Conceptualization, H.-J.S.; Methodology, H.-J.S.; Software, Y.-R.T. and W.-C.L.; Validation, H.-J.S., C.-T.Y. and W.-C.L.; Formal analysis, C.-T.Y.; Investigation, C.-M.L.; Data curation, Y.-R.T.; Writing—original draft, H.-J.S. and Y.-R.T.; Writing—review & editing, C.-T.Y.; Supervision, C.-M.L.; Project administration, C.-M.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Institutional Review Board Statement: Not Available.

Informed Consent Statement: Not Available.

Data Availability Statement: Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Amin, M.; Al-Obeidat, F.; Tubaishat, A.; Shah, B.; Anwar, S.; Tanveer, T.A. Cyber security and beyond: Detecting malware and concept drift in AI-based sensor data streams using statistical techniques. *Comput. Electr. Eng.* **2023**, *108*, 108702. [CrossRef]
2. Tariq, N.; Asim, M.; Al-Obeidat, F.; Zubair Farooqi, M.; Baker, T.; Hammoudeh, M.; Ghafir, I. The Security of Big Data in Fog-Enabled IoT Applications Including Blockchain: A Survey. *Sensors* **2019**, *14*, 1788. [CrossRef] [PubMed]
3. Al-Obeidat, F.; Hani, A.B.; Adedugbe, O.; Majdalawieh, M.; Benkhelifa, E. The Socio-economic Impacts of Social Media Privacy and Security Challenges. In Proceedings of the Frontiers in Cyber Security, Tianjin, China, 4 November 2020; pp. 553–563.
4. Bhatia, V.; Ramkumar, K.R. An Efficient Quantum Computing technique for cracking RSA using Shor's Algorithm. In Proceedings of the IEEE 5th International Conference on Computing Communication and Automation (ICCCA), Greater Noida, India, 30–31 October 2020.
5. Grover, L.K. A fast quantum mechanical algorithm for database search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, New York, NY, USA, 22–24 May 1996.
6. Grassl, M.; Langenberg, B.; Roetteler, M.; Steinwandt, R. Applying Grover's algorithm to AES: Quantum resource estimates. In Proceedings of the International Conference on Post-Quantum Cryptography, Fukuoka, Japan, 24–26 February 2016.
7. NIST. *Announcing the Advanced Encryption Standard (AES)*; Federal Information Processing Standards Publication: Gaithersburg, MD, USA, 2001; pp. 5–26.
8. Jose, L.H.; Guido, P.; Christof, E.; Mario, P. Quantum Computing. *IEEE Softw.* **2021**, *38*, 7–15.
9. Mattsson, J.P.; Smeets, B.; Thormarker, E. Quantum-Resistant Cryptography. *arXiv* **2021**, arXiv:2112.00399.
10. Al-Ghamdi, A.B.; Al-Sulami, A.; Aljahdali, A.O. On the security and confidentiality of quantum key distribution. *Secur. Priv.* **2020**, *3*, e111. [CrossRef]
11. Balupala, H.K.; Rahul, K.; Yachareni, S. Galois Field Arithmetic Operations using Xilinx FPGAs in Cryptography. In Proceedings of the IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), Toronto, Canada, 14 May 2021.
12. Feistel, H. Cryptography and Computer Privacy. *Sci. Am.* **1973**, *228*, 15–23. [CrossRef]

13. Webster, A.F.; Tavares, S.E. On the Design of S-Boxes. In *Advances in Cryptology—CRYPTO '85 Proceedings*; Springer: Berlin/Heidelberg, Germany, 1985; pp. 523–534.
14. Khose, P.N.; Raut, V.G. Implementation of AES algorithm on FPGA for low area consumption. In Proceedings of the International Conference on Pervasive Computing (ICPC), Pune, India, 8–10 January 2015.
15. Moh'd, A.; Jararweh, Y.; Tawalbeh, L.A. AES 512: 512-bit Advanced Encryption Standard algorithm design and evaluation. In Proceedings of the International Conference on Information Assurance and Security (IAS), Melacca, Malaysia, 5–8 December 2011.
16. D'souza, F.J.; Panchal, D. Advanced encryption standard (AES) security enhancement using hybrid approach. In Proceedings of the International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, India, 5–6 May 2017.
17. Hu, F.; Ni, F. Software Implementation of AES-128: Side Channel Attacks Based on Power Traces Decomposition. In Proceedings of the 2022 International Conference on Cyber Warfare and Security (ICCWS), Islamabad, Pakistan, 7–8 December 2022.
18. Sanap, S.D.; More, V. Performance Analysis of Encryption Techniques Based on Avalanche effect and Strict Avalanche Criterion. In Proceedings of the International Conference on Signal Processing and Communication, Coimbatore, India, 13–14 May 2021.
19. Sunil, J.; Suhas, H.S.; Sumanth, B.K.; Santhameena, S. Implementation of AES Algorithm on FPGA and on software. In Proceedings of the IEEE International Conference for Innovation in Technology (INOCON), Bangluru, India, 6–8 November 2020.
20. Singh, K.; Dod, S. An Efficient Hardware design and Implementation of Advanced Encryption Standard (AES) Algorithm. *Comput. Sci.* **2016**. [CrossRef]
21. Kumar, K.; Singh, V.; Mishra, G.; Babu, B.R.; Tripathi, N.; Kumar, P. Power-Efficient Secured Hardware Design of AES Algorithm on High Performance FPGA. In Proceedings of the 2022 5th International Conference on Contemporary Computing and Informatics (IC3I), Uttar Pradesh, India, 14–16 December 2022.
22. Rahim, U.; Siddiqui, M.F.; Javed, M.A.; Nafi, N. Architectural Implementation of AES based 5G Security Protocol on FPGA. In Proceedings of the 2022 32nd International Telecommunication Networks and Applications Conference (ITNAC), Wellington, New Zealand, 30 November–2 December 2022.
23. Shet, G.G.; Jamuna, V.; Shravani, S.; Nayana, H.G.; Kumar, P. Implementation of AES Algorithm Using Verilog. *JNNCE J. Eng. Manag.* **2020**, *4*, 1. [CrossRef]
24. Soumya, V.H.; Neelagar, M.B.; Kumaraswamy, K.V. Designing of AES Algorithm using Verilog. In Proceedings of the International Conference for Convergence in Technology (I2CT), Mangalore, India, 27–28 October 2018.
25. Srinivas, N.S.; Akramuddin, M.D. FPGA based hardware implementation of AES Rijndael algorithm for Encryption and Decryption. In Proceedings of the International Conference on Electrical Electronics and Optimization Techniques (ICEEOT), Chennai, India, 3–5 March 2016.
26. Shah, S.S.; Raja, G. FPGA implementation of chaotic based AES image encryption algorithm. In Proceedings of the IEEE International Conference on Signal and Image Processing Applications (ICSIPA), Kuala Lumpur, Malaysia, 19–21 October 2015.
27. Joshi, A.; Dakhole, P.K.; Thatere, A. Implementation of S-Box for Advanced Encryption Standard. In Proceedings of the IEEE International Conference on Engineering and Technology (ICETECH), Coimbatore, India, 20 March 2015.
28. Nadjia, A.; Mohamed, A. Efficient implementation of AES S-box in LUT-6 FPGAs. In Proceedings of the International Conference on Electrical Engineering (ICEE), Boumerdes, Algeria, 13–15 December 2015.
29. Pammu, A.A.; Chong, K.S.; Ne, K.Z.; Gwee, B.H. High Secured Low Power Multiplexer-LUT Based AES S-Box Implementation. In Proceedings of the International Conference on Information Systems Engineering (ICISE), Los Angeles, CA, USA, 20–22 April 2016.