Advanced Cryptographic System for data Encryption and Decryption

Suresh M G
Electronics and Communication Engineering
RJIT –Bangalore/ VTU- Belgaum
India

Dr. Nataraj K.R
Electronics and Communication Engineering
SJBIT-Bangalore/ VTU- Belgaum,
India

***Abstract*- in this paper, we mainly focus on the implementation of Advanced Cryptographic System using two crypto algorithms in a single chip to provide high security and high performance. This paper proves the confidentiality of data over insecure medium. The two algorithms namely 128-bit AES and RSA implemented in a single chip prove difficult for the hacker to crack the system. It combines two transformations of AES algorithms which achieves high speed and less area on chip. This system generates keys internally, which also achieves high security and high performance with faster execution of the algorithm.**

Keywords- AES, RSA, Cryptographic system, Key.

I. INTRODUCTION

Cryptography is the fundamental component for securing the Internet traffic. However, cryptographic algorithms impose tremendous processing power demands that can be a bottleneck in high-speed networks. The implementation of a cryptographic algorithm must achieve high processing rate to fully utilize the available network bandwidth. To follow the variety and the rapid changes in algorithms and standards, a cryptographic implementation must also support different algorithms and be upgradeable in field. Otherwise, interoperability among different systems is prohibited and any upgrade results in excessive cost. The ultimate solution for the problem would be an adaptive processor that can provide software-like flexibility with hardware-like performance.

Before the modern era, cryptography is concerned only with message confidentiality (i.e., encryption) means conversion of comprehensible form of information into an incomprehensible one. Encryption was used to ensure security in communications, such as military applications. In recent decades, the field has expanded beyond confidentiality concerns to include techniques for message integrity checking, sender/receiver identity authentication, digital signatures, and interactive proofs and secure computation among others.

Cryptography is used for confidentiality, authentication, data integrity, and non-repudiation, which can be divided into two families: secret-key cryptography and public-key cryptography. Secret-key cryptography which usually has a relatively compact architecture and smaller key size than public-key cryptography is often used to encrypt/decrypt sensitive information or documents. Public-key cryptography offers fundamental technology for key agreement, encryption/decryption (two keys), and digital signatures. In contrast to secret-key cryptography, public-key cryptography usually has a lower throughput rate and more complicated computation. Modern cryptography intersects the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce.

Cryptography offers a robust solution for IT security by providing security services in terms of confidentiality, data integrity, authenticity and non-repudiation. These services form the core operations in Public Key Infrastructure (PKI), which is an essential framework for managing digital certificates and encryption keys for people, programs and systems [1][2]. PKI-enabling functions are required in secure electronic systems applied in e-commerce, ehealth systems, e-government (e.g. *MyKad)* and secure military communications. The crypto processor proposed in this work supports comprehensively these functions, which include 128-bit AES, 163-bit ECC, 1024-bit RSA, LZSS data compression, SHA-1 hashing, wide-operand modular arithmetic. AES (Advanced Encryption Standard) is replacing DES and 3DES as the standard for private key cryptography. RSA (Rivest, Shamir & Adleman is currently the legacy and widely installed public key cryptography, which will migrate to ECC (Elliptive Curve Cryptography) in next-generation security devices [3].
During the selection process of the AES [4], an important criterion was the efficiency of the cipher in different platforms, including FPGAs. Since 2001, various implementations have consequently been proposed, exploring the different possible design tradeoffs ranging from the highest throughput to the smallest area [5]. Each of those implementations usually focuses on a particular understanding of "efficiency". The three major design targets with respect to hardware realization are: optimization for area or cost, low latency that minimizes time to encrypt a single block and high throughput to encrypt multiple blocks in parallel. All these design criteria involve a trade-

off between area and speed. There is a wide range of equipment where encryption is needed for authentication and security but throughput is not the principal concern. A low cost, small area design could be used in smart card applications as well as in other storage devices and low speed communication channels [6].

RSA relies heavily on complex large-number mathematics to provide its security services. Computationally intensive software, typically VPN applications, is used for computer-based RSA cryptography resulting in less than adequate communication performance. This can be overcome by using dedicated ASIC or ASSPs to accelerate the mathematics, but these are often expensive and inflexible as a solution. The combined cost and performance problem can be addressed by considering an FPGA based implementation. For this, many research papers propose many different solutions, none of which to date are viable for practical implementations in FPGAs. To achieve realistic hardware implementations for RSA, the complex math involved utilizes a technique known as Montgomery Multiplication. Montgomery's techniques allow very efficient implementations of RSA-based cryptography systems. The calculations involved with Montgomery are based around the cyclic re-use of additions and the challenges faced with FPGA implementations centre around this [7]. The fundamental operation of the algorithm is modular exponentiation which is achieved by repeated modular multiplications. The Montgomery modular multiplication algorithm is often used to perform these calculations. However, the high bit lengths required to provide adequate security (1024 bits is considered secure against attack in the near future), mean a high hardware throughput is difficult to achieve. An efficient algorithm for the calculation of (Ax B) mod M was developed by P. L. Montgomery and forms the basis of the designs presented here. It should be noted that Montgomery's algorithm only works if the modulus is relatively prime to the radix, although this is always the case in RSA [8].

II. AES ALGORITHM

AES algorithm is a symmetric block cipher that can encrypt and decrypt information. Encryption converts data to an unintelligible form called cipher-text. Decryption of the cipher-text converts the data back into its original form, which is called plaintext. The AES algorithm is capable of using cryptographic keys of 128, 192, and 256 bits. The sizes of the data blocks and keys can be chosen independently. In this paper, 128-bits for both I/O block and user keys are assumed. Therefore the cipher in all configuration presented operates in Nr=10 rounds.

Fig.1 shows the encryption and decryption structure of AES algorithm. After the initial round key operation, Nr rounds are performed. The first Nr-1 rounds are same, with a small difference in final round as shown in Fig.1 (a), the first Nr-1 rounds mainly consists of 4 transformations: Sub Bytes, Shift Rows, Mix Columns and Add Round Key. The final round excludes the Mix Column transformation. The decryption algorithm uses inverse forms of the transformations used in the encryption algorithm as shown in Fig.1 (b).

The National Institute of Standards and Technology, (NIST), solicited proposals for the Advanced Encryption Standard, (AES). The AES is a Federal Information Processing Standard, (FIPS), which is a cryptographic algorithm that is used to protect electronic data. The AES algorithm is a symmetric block cipher that can encrypt, (encipher), and decrypt, (decipher), information. Encryption converts data to an unintelligible form called cipher-text.

Decryption of the cipher-text converts the data back into its original form, which is called plaintext. The AES algorithm is capable of using cryptographic keys of 128, 192, and 256 bits to encrypt and decrypt data in blocks of 128 bits. Many algorithms were originally presented by researchers from twelve different nations. Fifteen, (15), algorithms were selected from the first set of submittals. After a study and selection process five, (5), were chosen as finalists. The five algorithms selected were MARS, RC6, RIJNDAEL, SERPENT and TWOFISH. The conclusion was that the five Competitors showed similar characteristics. On October 2nd 2000, NIST announced that the Rijndael Algorithm was the winner of the contest. The Rijndael Algorithm was chosen since it had the best overall scores in security, performance, efficiency, implementation ability and flexibility, [NIS00b].

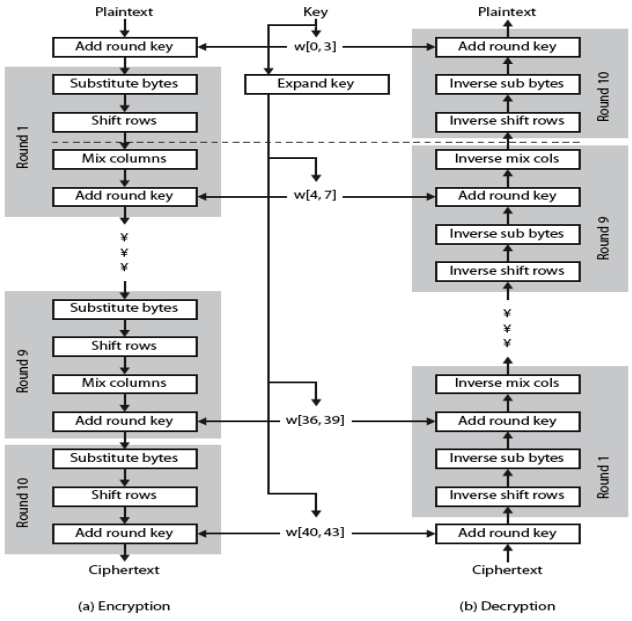

Fig.1. Structure of AES

A. Add Round Key

Add Round Key is an XOR between the state and the round key. This transformation is its own inverse.

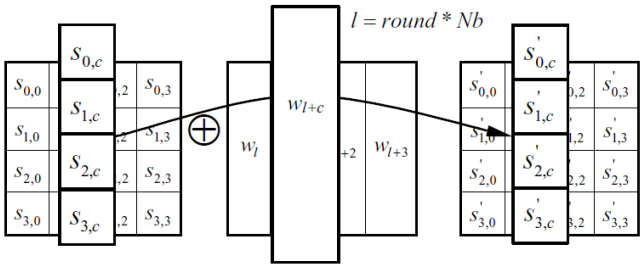

Fig. 2. Add Round Key Transformation

B. SubBytes

Sub Bytes is a substitution of each byte in the block independent of the position in the state

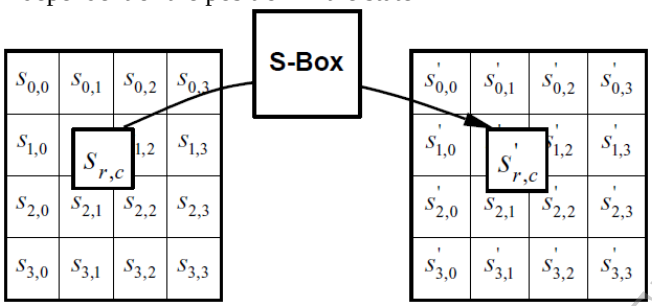

Fig. 3. Sub Byte Transformation

This is an S-box. It is a bijection on all possible byte values and therefore invertible (the inverse S-box can easily be constructed from the S-box). This is the non-linear transformation. The S-box used is proved to be optimal with regards to non-linearity. The S-box is based on arithmetic in GF (28).

C.ShiftRows

Shift Rows is a cyclic shift of the bytes in the rows in the state and is clearly invertible (by a shift in the opposite direction by the same amount).

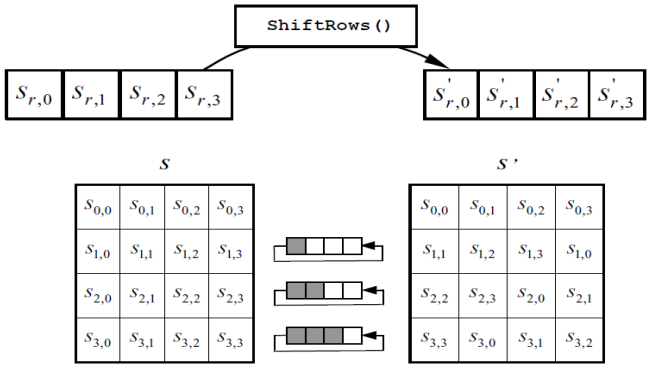

Fig. 4. Shift Rows Transformation

D. Mix Columns

Each column in the state is considered a polynomial with the byte values as coefficients. The columns are transformed independently by multiplication with a special polynomial c(x). c(x) has an inverse d(x), that is used to reverse the multiplication by c(x).

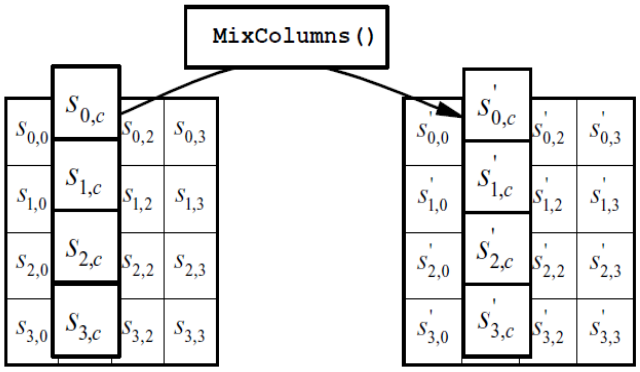

Fig.5. Mix Columns Transformation

The decryption model given in Fig. 1(b) is modified from the original described in the NIST standard. It has been rearranged with some changes to the key generation unit to obtain a structure similar to the encryption model. This modification is based on the properties of the AES algorithm, which is:

- *InvShiftRows* transformation immediately followed by an *InvSubBytes* transformation is equivalent to *InvSubBytes* transformation immediately followed by an *InvShiftRows* transformation.
- *InvMixColumns* transformation is linear, which means:
  InvMixColumns (State XOR roundkey) = *InvMixColumns* (State) XOR *InvMixColumns* (roundkey)

III. RSA ALGORITHM

The RSA algorithm, invented in 1977 by Rivest, Shamir, and Adleman, is currently the most popular public key cryptosystem in use, particularly in high-end commercial software products that are typically employed in ecommerce and VPN servers. It can provide encryption and digital signatures. In hardware implementations, the
RSA algorithm can be found in secure telephones, in Ethernet network cards, and smart cards. The main advantage of the algorithm is that it can provide both data confidentiality service (via public-key encryption) and data integrity, authentication and non-repudiation (via digital signatures) using the same key pair and under the same mathematical operation. Its hard problem is based on the large integer factorization problem [9].

RSA algorithm is the essence of simplicity [9]. To encrypt a message X to its cipher text P, we perform $P = X^E \bmod M$ using the public key (E, M). To restore the message, $X = P^D \bmod M$ is performed, where (D, M) is the private key. For digital signature purpose, we use the private key in signing, $S = X^D \bmod M$. To verify the signature, we use the public key to perform $X = S^E \bmod M$. Fig. 6 below shows the RSA process. In RSA, whether encrypting, decrypting, signing, or verifying, the operation is basically a wide-operand modular exponentiation. The basic modular exponentiation equation, given by: $Y = X^E \ mod \ M$, essentially consists of thousands of modular multiplication, *A.B mod M*, in *GF (p)*. Thus, the modular exponentiation is computation-intensive and requires a long computation time. A proven method to speed up its operation time is to utilize Montgomery modular multiplications, which do not have divisions.

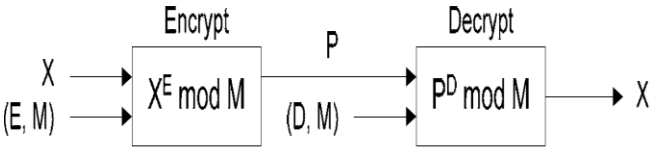

Fig. 6: RSA Operation

The Modular Exponentiation algorithm, *ModExp ( )*, applied in this work is given in Algorithm 1, Fig. 7 below. It utilizes the Montgomery modular multiplication, *MonMult ( )*, with the algorithm given in Fig. 8. For hardware implementation, the multi-precision version of Montgomery multiplication is employed, in which the algorithm assumes that M is an m-digit number in radix-r, and $R = r^m$. In our hardware design, radix-2 is chosen.

In RSA, each user has a pair of keys, i.e. public key *(E, M)* and private key *(D, M)*. *M* is the modulus, *E* is the public exponent and *D* is the private exponent. RSA key length refers to the bit length of the modulus. These keys are generated using the algorithm as shown in Fig.7. *P and q* must be kept secret or deleted. (*gcd* stands for greatest common divisor. *D* is the multiplicative inverse of *E* modulo *M*.).

1. Generate 2 primes, p and q randomly.
2. Calculate $M = p.q$, and $\Phi(M) = (p-1).(q-1)$.
3. Generate E that fulfills both the conditions $1 < E < \Phi(M)$, and $gcd\,(\Phi(M), E) = 1$.
4. Calculate $D = E^{-1} \ mod \ \Phi(M)$.

Fig. 7: RSA key Pair Generation Algorithm

Algorithm-1:
ModExp(R^2 mod M, X, E, M)

Compute $P = X^E \bmod M$,

$$E = \sum_{i=0}^{n-1} e_i \cdot 2^i \ , \ e_i \in \{0, 1\}$$

1. $P_0 = \textbf{\textit{MonMult}}(1, R^2 \bmod M, M)$
   $Z_0 = \textbf{\textit{MonMult}}\ (X, R^2 \bmod M, M)$
2. For $i = 0$ to n-1
2a. $P_{temp} = \textbf{\textit{MonMult}}(P_i, Z_i, M)$
   $Z_{i+1} = \textbf{\textit{MonMult}}(Z_i, Z_i, M)$
2b. if $e_i = 1$ then $P_{i+1} = P_{temp}$
   else $P_{i+1} = P_1$
3. End For
4. $P = \textbf{\textit{MonMult}}\ (P_n, 1, M)$

Modular Exponentiation algorithm

Algorithm-2: *MonMult(A, B, M)*

Compute $P = A.B.2^{-(m+2)} \ mod \ M$,

$$M = \sum_{i=0}^{m-1} m_i \cdot 2^i$$

$$B = \sum_{i=0}^{m} b_i \cdot 2^i \ , \ B < 2M$$

$$A = \sum_{i=0}^{m+2} a_i \cdot 2^i \ , \ m_i, b_i, a_i \in \{0, 1\},$$
$$a_{m+1} = a_{m+2} = 0, \ A < 2M$$

1. $P_0 = 0$
2. For $i = 0$ to m+2 Loop
2a. $q_i = p_{i,0}$
2b. $P_{i+1} = (P_i + q_i \cdot M)/2 + a_i \cdot B$
3. End Loop
4. Return $P = P_{m+3}$

Montgomery Modular Multiply

Fig.3 it shows TMS320C6x, TI (Texas Instruments) which supports operations like shuffling and de-shuffling of two input words and the extraction operation takes place depending on the offset [9]. Figure 4 it shows TMS320C55x it supports expand and extract instructions [10].

VI. ADVANCED CRYPTOGRAPHIC SYSTEM

Fig.8 shows the advanced cryptographic system. System 1 sends some original message to AES, RSA encrypted Output as a key to AES, In RSA Original AES key here acts a RSA Data and we give Public key to RSA, RSA Performs its operation and gives encrypted output to AES. The AES algorithm generates a fresh symmetric key for the message to be encapsulated. It encrypts the message using the symmetric key just generated. The symmetric key for AES is encrypted by RSA, using System 2 public key. Under this, The System 2 uses private key to decrypt the symmetric key present in RSA. The obtained key is used to further decrypt the message in AES. The decrypted message as an original data of System1. And the simulation results are shown in fig 9, Fig10 and Fig11,

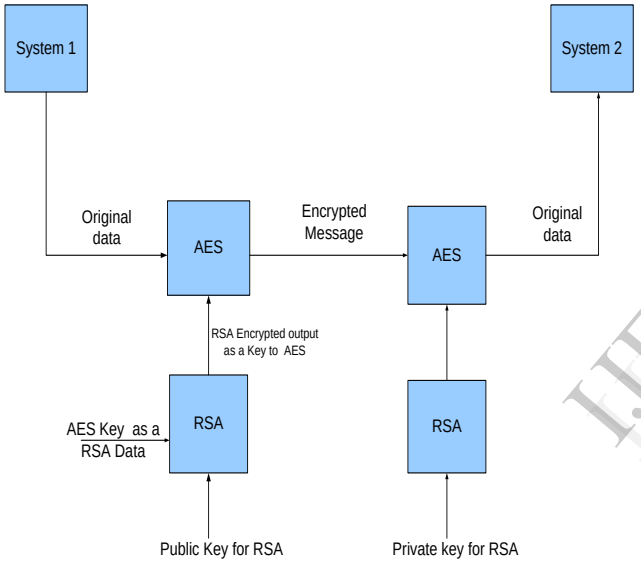

Fig.8. Proposed System Overview.

V. RESULTS

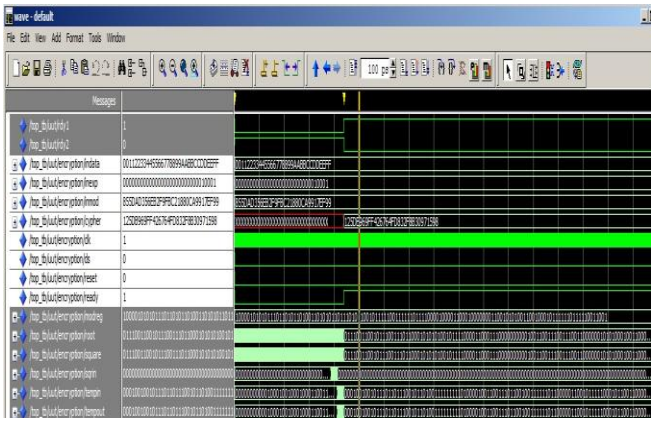

Fig 9. RSA Encryption Simulation

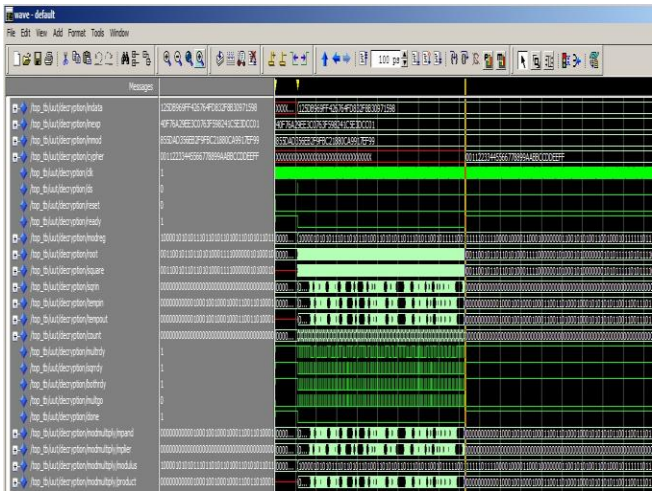

Fig10. RAS Decryption Simulation

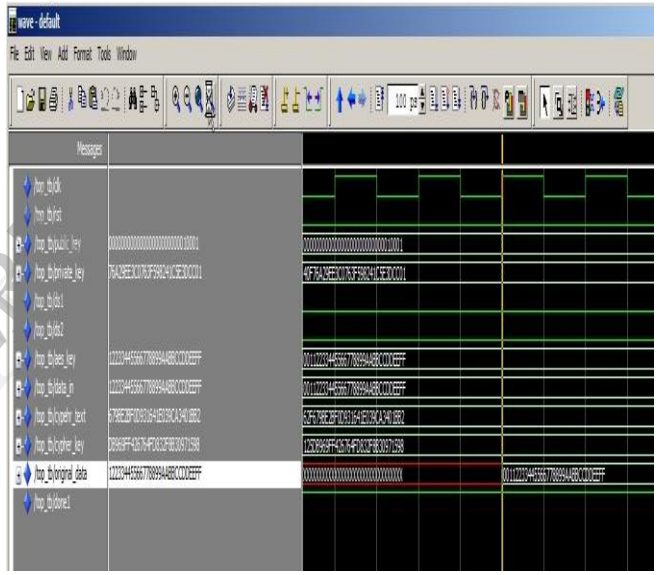

Fig11.Final Simulation Results

IV. CONCULSION

We proposed the crypto system consists of set of crypto processors perform two security schemes which is AES encryption scheme and RSA scheme. The former is to provide privacy and confidentiality of message while the latter provides security services in terms of data integrity, authentication and non-repudiation. Cryptographic system is useful for security purpose in various platforms. The developed system will be highly effective to protect any kind of data. AES and RSA algorithms are effective algorithms used in cryptographic system. The dual core comprises of AES and RSA algorithms on a single chip with switching between them, which provides much complex system to be cracked.

REFERENCES

[1] P. C. van Oorschot, A. J. Menezes, and S.A. Vanstone, "Handbook of Applied Cryptography", *CRC press Inc.,Florida*, 1996.

[2] Understanding Public Key Infrastructure, RSA Data Security, 1999.

[3] The Elliptic Curve Cryptosystem, Certicom Corp., July 2000.

[4] National Institute of Standards and Technology. Advanced Encryption Standard (AES). Federal Information Processing Standards Publications –FIPS 197,

[5] K. J¨arvinen, M. Tommiska, and J. Skytt¨a. Comparative survey of highperformance cryptographic algorithm implementations on FPGAs. In IEE Proceedings on Information Security, volume 152, pages 3 – 12, October 2005

[6] M. C. Liberatorl and j. C. Bonadero "Aes-128 Cipher. Minimum Area, Low Cost FPGA Implementation"

[7] John Fry, Martin Langhammer,"RSA & Public Key Cryptography in FPGAs", *Altera Corporation – Europe.*

[8] Ridha Ghayoula, ElAmjed Hajlaoui,"FPGA Implementation of RSA Cryptosystem"

[9] *Mohamed Khalil Hani, Hau Yuan Wen, Arul Paniandi,"* design and implementation of a private and public key crypto processor for next-generation it security applications".