International Journal of New Technology and Research (IJNTR)
ISSN: 2454-4116, Volume-5, Issue-6, June 2019 Pages 23-26

Cryptography using Modular Arithmetics

Shrivathsa B, Nisha, Kiran Kumar V G

***Abstract*— The image is encrypted and decrypted using modular operations such as addition and multiplication. The operations like rotation and ex-or implemented to increase the complexity of the encryption algorithms. All these implemented and simulated in Xilinx ISE. Area, delay of encryption and decryption are tabulated and compared.**

***Index Terms*— ISE, cipher text, plain text, Montgomery.**

I. Introduction

Highlight The usage of laptops, mobiles and other portable devices has been increased dramatically over last decades due to advancement in the VLSI field. With increase in the demand for portable devices, engineers around the world are developing powerful devices with reducing power consumption. The power consumption and portability are key factor of wireless devices. The aim of design to make device with less power dissipation, less area, less delay. The data communication between two entity are encrypted so that third person cannot able to get the data without knowing key.

Encryption is procedure of changing over the plain content into another structure key with the goal that the information can not be gotten to by some other than authorized device. The motivation behind encryption is to secure the secrecy of advanced information put away on computerized frameworks or to ensure information which are transmitted. The cutting edge encryption has following key components of security

• Authentication: the inventiveness of a message can be checked

•Integrity: verification that the substance of a message not changed

• Privacy/secrecy: ensures no one can peruse message

• Non-denial: the sender can't deny sending the message

In its most punctual structure, individuals have been endeavouring to conceal certain data that they needed to keep to their very own belonging by substituting portions of the data with images, numbers and pictures. Egyptian, Aryans, Greeks, Spartans were utilizing encryption to ensure their messages are safe.

In paper [1] & [2], the author has discussed about cryptography, encryption, decryptions and RSA public key methods which are commonly used. They also given various mathematic algorithms for Public-key cryptographic systems and the implementation of RSA algorithm in detail.

Xiaodong Yan and Shuguo Li [2] described a modular inversion algorithm that computes the inverse modular of a 256bit input. Compared to other algorithms, the reduction in operation cycles is 31%. It is suitable in kernel computing engines used in high speed cryptography.

In paper [3],[4], implemented Montegomery multiplication and Modular exponential method for encryption and decrypting the required data.

In paper [5], used logical operations like AND, OR, EXOR for encrypting the images in FPFA

II. IMPLIMENTATION

A. Montgomery multiplication and reduction:

Montgomery multiplication is used for modular multiplication operation. The Montgomery algorithm for modular multiplication is considered to be the fastest algorithm to compute xy mod n in computers when the values of x, y, and n are large. It requires Montgomery reduction to get multiplication value. In this experiment this is used for modular operation by making y=1. i.e. x mod n.

Table 1: Pseudo code for Montgomery multiplication

```
Inputs:
  x=(xk−1xk−2…x2x1x0)
  y=(yk−1yk−2…y2y1y0)
  p=(pk−1pk−2… p2p1p0)

Steps:
  p′= −p−1 mod b
  t=0 ( t=tn−1tn−2t2t1t0)
  for i=0 to k−1
      a=((t0+($x_i$∗ y0)) p′) mod b
      t=(t+$x_i$y+$a_i$p)/ b
  while t≥p , t=t−p

xyR−1 mod p=t
```

Table 2: Pseudo code for Montgomery reduction

```
Inputs:
  t=(tk−1tk−2… t2t1t0)
  p=(pk−1pk−2… p2p1p0)

Steps:
  p′= −p−1 mod b
  a=t ( a=ak−1ak−2a2a1a0)
  for i=0 to k−1
      $u_i$= $a_i$p′ mod b
      a=a+ $u_i$p$b^i$
      a=a/bk
  while a≥p , a=a−p

tR−1 mod p=a
```

Shrivathsa B, M.tech student, EC department, SCEM, Adyar
Nisha, Assist. Professor, EC Department, SCEM, Adyar
Kiran Kumar V G, Associate Professor, Ec Deptarment, SCEM, Adyar

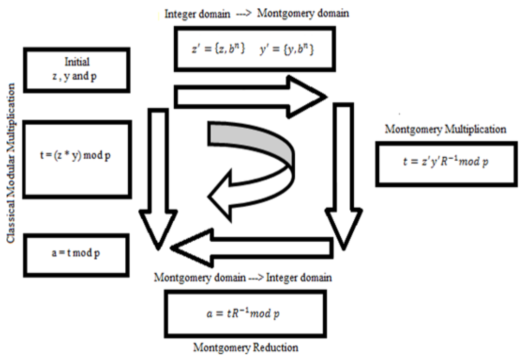
Integer domain ---> Montgomery domain
$z' = \{z, b^n\}$ $y' = \{y, b^n\}$
Initial
z , y and p
Classical Modular Multiplication
t = (z * y) mod p
a = t mod p
Montgomery Multiplication
$t = z'y'R^{-1} mod\ p$
Montgomery domain ---> Integer domain
$a = tR^{-1} mod\ p$
Montgomery Reduction

Figure 1: Flowchart of Montgomery modular algorithm

B. Modular addition Cryptography:

In encryption, modular addition implemented using Montgomery method which is then rotated by 4 bits and exor-ed to get cipher text. While decrypting the cipher text is subtracted with key in modular operation followed by rotating bits and exoring**.**

Table 3: Pseudo code for encryption by modular addition

```
Inputs:
$a=(a_{k-1}\ a_{k-2}\dots a_2a_1a_0)$
$b=(a_{k-1}b_{k-2}\dots b_2b_1b_0)$ //key
$m=(m_k-1m_{k-2}\dots m_2m_1m_0)$
$k=(k_k-1k_{k-2}\dots k_2k_1k_0)$
Output:
$y=(y_{k-1}y_{k-2}\dots y_2y_1y_0)$
Steps:
$z=(z_k-1z_{k-2}\dots z_2z_1z_0)$
z=(a+b)%m
z<<4
y=z^k
```

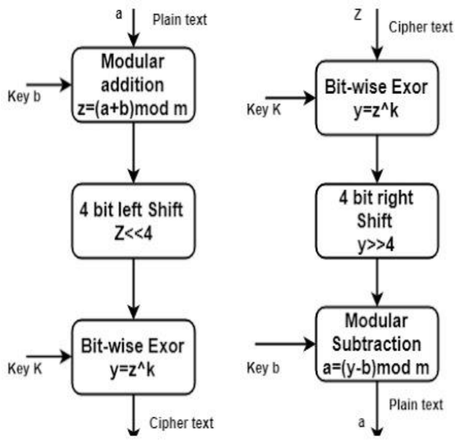

Figure 2: Flowchart of encryption and decryption using modular addition

Table 4: Pseudo code for decryption by modular addition

```
Inputs:
$z=(z_k-1z_{k-2}\dots z_2z_1z_0)$
$b=(a_{k-1}b_{k-2}\dots b_2b_1b_0)$ //key
$m=(m_k-1m_{k-2}\dots m_2m_1m_0)$
$k=(k_k-1k_{k-2}\dots k_2k_1k_0)$
Output:
$y=(y_{k-1}y_{k-2}\dots y_2y_1y_0)$
Steps:
$c=(c_k-1c_{k-2}\dots c_2c_1c_0)$
c=z^k
c>>4
d=(z-b)%m
if(d<-1)
        d=d+m
$y=d(d_k-1d_{k-2}\dots d_2d_1d_0)$
```

C. Modular multiplication cryptography:

In order to perform modular multiplication, montgomery multiplication method is used since it works much faster than normal modular multiplication process. The multiplicative product is then shifted 4 bit and exor-ed to get final cipher value. While decrypting these values, it is exor, shifted and the inverse of data is multiplied with key and mod of inverse of this multiplication product is performed.

Table 5: Pseudo code for encryption by modular multiplication

```
Inputs:
clk
$x=(x_{k-1}x_{k-2}\dots x_2x_1x_0)$
$y=(y_{k-1}y_{k-2}\dots y_2y_1y_0)$
$m=(m_{k-1}m_{k-2}\dots m_2m_1m_0)$
$k=(k_{k-1}k_{k-2}\dots k_2k_1k_0)$
Output:
$y=(y_{k-1}y_{k-2}\dots y_2y_1y_0)$
Steps:
z=(x*y) mod m
z<<4
y=z^k
```

Table 6: Pseudo code for decryption by modular multiplication

```
Inputs:
clk
$z=(z_{k-1}z_{k-2}\dots z_2z_1z_0)$
$b=(b_{k-1}b_{k-2}\dots b_2b_1b_0)$
$m=(m_{k-1}m_{k-2}\dots m_2m_1m_0)$
$k=(k_{k-1}k_{k-2}\dots k_2k_1k_0)$
Output:
$y=(y_{k-1}y_{k-2}\dots y_2y_1y_0)$
Steps:
 $t=(t_{n-1}t_{n-2}t_2t_1t_0)$
t=z^b
t>>4
$y=[(t^{-1})b]^{-1}$ mod m
```

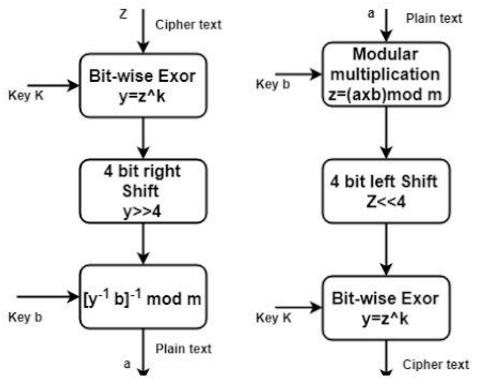

Figure 3: Flowchart of encryption and decryption using modular addition

III. SIMULATION

Verilog code of cryptography using modular operation is synthesized on Xilinx ISE Design Suite 14.2 Here testbench waveform and device utilization summary are shown which are generated using Xilinx software.

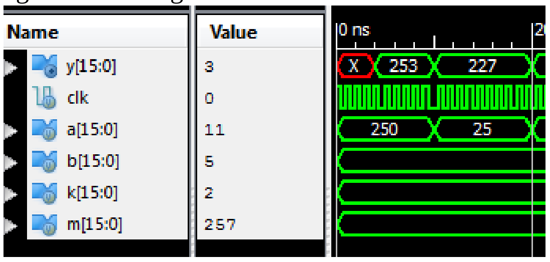

Figure 4a: Waveform of encryption using modular addition

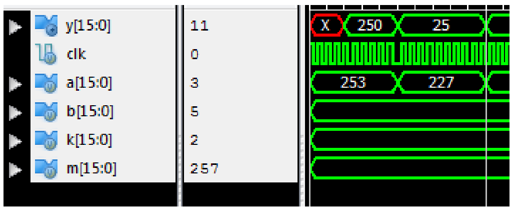

Figure 4b: Waveform of decryption using modular addition

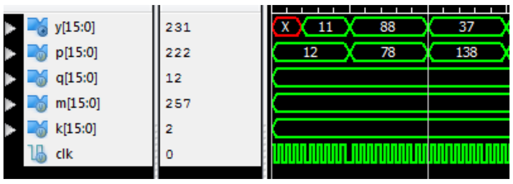

Figure 5a: Waveform of encryption using modular multiplication

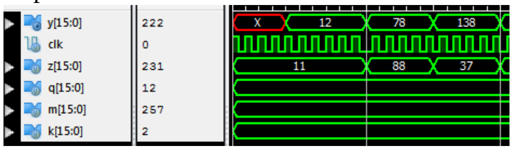

Figure 5b: Waveform of decryption using modular multiplication

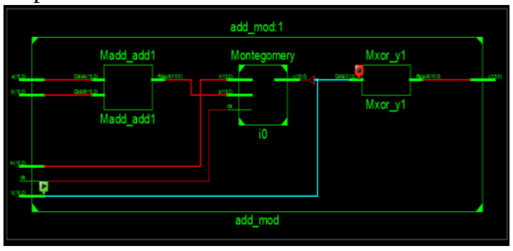

Figure 6a: RTL schematic of encryption using modular addition

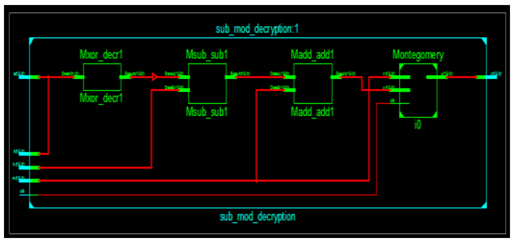

Figure 6b: RTL schematic of decryption using modular addition

Table 7: Ti:ming and area report

Vertex 6				
Modular Addition	Delay (ns)	Slice register	Slice LUT	IOB's
Encryption	105.26	71	4075	81
Decryption	105.43	71	4104	81

Table 8: Timing and area report

Vertex 6				
Modular Multiplicat--ion	Delay (ns)	Slice register	Slice LUT	IOB's
Encryption	107.64	71	4230	81
Decryption	106.72	134	8008	81

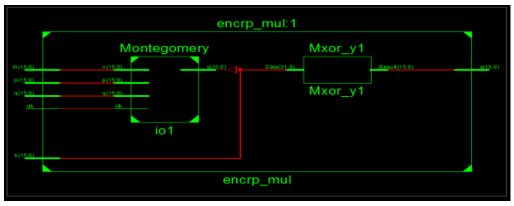

Figure 7a: RTL schematic of encryption using modular multiplication

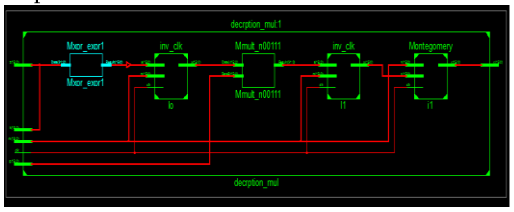

Figure 7b: RTL schematic of decryption using modular multiplication

IV. CONCLUSION

Each of the algorithms play a very important role in many complex applications like signal processing, data processing, etc. In addition, even though the area and the timing constraints of both the algorithms is same in Vertex6, Even though bitwise exor multiplication is used in cryptography, Modular multiplication using montgomery multiplication appears to be much better than modular addition.

The selection of algorithm is done by designer based on values of constraints i.e. area, power and timing. An algorithm that requires larger area but is the fastest cannot be used in an application in which area is critical. The utilization of an algorithm depends on the application. Usage of all the implemented algorithms in cryptographic algorithms like RSA, AES and so on can be done to design a more secure and less complex system.

REFERENCES

[1] Zhou, Xin, and Xiaofei Tang. "Research and implementation of RSA algorithm for encryption and decryption." In *Proceedings of 2011 6th International Forum on Strategic Technology*, vol. 2, pp. 1118-1121. IEEE, 2011.

[2] Yan, Xiaodong, and Shuguo Li. "Modified modular inversion algorithm for vlsi implementation." In *2007 7th International Conference on ASIC*, pp. 90-93. IEEE, 2007.

[3] Šimka, Martin, Viktor Fischer, and Miloš Drutarovský. "Hardware-Software Codesign in Embedded Asymmetric Cryptography Application–a Case Study." In *International Conference on Field Programmable Logic and Applications*, pp. 1075-1078. Springer, Berlin, Heidelberg,2003.

[4] Daly, Alan, and William Marnane. "Efficient architectures for implementing montgomery modular multiplication and RSA modular exponentiation on reconfigurable logic." In *Proceedings of the 2002 ACM/SIGDA tenth international symposium on Field-programmable gate arrays*, pp. 40-49. ACM, 2002.

[5] Leong, M. P., Siti Zarina Md Naziri, and S. Y. Perng. "Image encryption design using FPGA." In *2013 International Conference on Electrical, Electronics and System Engineering (ICEESE)*, pp. 27-32.IEEE,2013