Reconfigurable Cryptographic Processor for Multiple Crypto-Algorithms

Rajesh Kannan Megalingam, Iype P Joseph, Gautham P, Parthasarathy R, Deepu K B, Mithun Muralidharan Nair
Amrita Vishwa Vidyapeetham, Kollam, Kerala, India.
rajeshm@am.amrita.edu, iype_joseph@hotmail.com, gautham.dondbz@gmail.com, partha_parthu@yahoo.co.in,
deepu13589@gmail.com, mithun34@ieee.org

Abstract— **In today's world there is a growing demand for real-time implementation of cryptographic algorithms which are being used in secure communication systems, networks and security systems. Traditional computing techniques involved the use of application specific integrated circuits to achieve high performance but with extremely inflexible hardware design meanwhile the flexibility of hardware design was achieved at the cost of slow processing by using general purpose processors. In this research paper a novel reconfigurable processor architecture has been presented for cryptographic applications that bridges the above mentioned gap and also sustains implementations that can show equal or even better performance results than custom-hardware and hitherto preserves all the flexibility of general-purpose processors. We present implementations for representative algorithms of block cipher such as Rijndael, RC5 and RC6 on our architecture. The RTL Description is done in ModelSim using Verilog HDL and the results are synthesized in Synopsys. This work presents an emerging reconfigurable hardware that potentially delivers flexible high performance for cryptographic algorithms.**

Keywords- cryptography; reconfigurable; datapath.

I. INTRODUCTION

Cryptographic services are required across variety of platforms in a wide range of applications such as secure access to private networks, electronic commerce and health care. Generally, most of cryptography algorithms are implemented in software, but software implementation cannot offer the physical security for the key. Software is operating system (OS) dependent and also exposed to viruses and hackers attacks that may interrupt the OS running on the general computer, for example on Microsoft Windows based computer or Apple Macintosh machine. Execution on general-purpose processor (CPU) of the algorithm will use most CPU's resources to calculate and execute all processes in the algorithm because CPU lacks of instructions for modular arithmetic with operations on very large operands. Thus, word sizes mismatch, less parallel computations and algorithm/architecture are the main problems faced by software implementation of cryptosystem.

II. MOTIVATION AND PROBLEM DEFENITION

Different applications of the data encryption algorithm may require different speed/area trade-offs. Some applications, such as smart card and cellular phone, require a small area. Other applications, such as World Wide Web (WWW) servers and Asynchronous Transfer Mode (ATM) networks are speed critical. Some other applications, such as digital video recorders, require an optimization of speed/area ratio. Problems faced by normal Cryptographic Processors include lack of adaptability to new encryption algorithms, unknown knowledge of the internal design, larger area, more power, huge manufacturing cost and less control to the user.

In general, hardware based solution are the embodiment of choice for military and serious commercial applications. As an encryption algorithm running on a generalized computer has no physical protection, hardware cryptographic devices can be securely encapsulated to prevent any modification of the implemented algorithm and also can be embedded the hardware as co-processor in any devices that require data security processing.

III. RELATED WORKS

Implementing public-key cryptosystems on a general-purpose processor (GPP) is flexible because a variety of cryptosystems can be used at runtime [1], [3]. A drawback of a GPP realization is that it generally results in a lower throughput rate and larger power consumption [2]. Considerable effort has been directed toward a fast realization of cryptography algorithms consisting of very large integer operands (up to 4096 bits). For real-time applications, a dedicated hardware implementation is required to speed up the computation of cryptosystems. In particular, an application-specific integrated circuit (ASIC) solution generally leads to a higher throughput rate at a lower cost, but it is inflexible [4], [5]; therefore, it is only applied to a limited subset of cryptosystems. To mitigate the gap between GPP and ASIC realization, application-specific cryptographic processors have been proposed, each with its own instruction-set Architecture, datapath design, and target applications. Moreover, microcode-based architectures can be adopted to program new or optimized cryptography operations in microcode read-only memories (ROMs) or lookup tables (LUTs) for enhancing the extensibility of cryptographic processors. The advantages of reconfigurable platforms included area and power efficiency, flexibility, algorithm upgradability, cost and resource efficiency and high throughput.

Research papers published on reconfigurable architecture for cryptographic processors are centered at implementing single cryptographic algorithm or a generic module suitable for few cryptographic algorithms [9], [12], [13]. The proposed

reconfigurable processor supports three algorithms namely AES, RC5 and RC6. As part of the research work, the Reconfigurable Cryptographic Processor design is implemented on hardware with key stored in a RAM, which can make not only a forward key scheduling for encryption but also a reversed key scheduling for decryption. Therefore, compared to software implementation, hardware implementation enhances the physical security as well as higher speed. Also intruders cannot easily attack, interrupt or modify its operation.

IV. Research Methodology, Techniques and Tools

As part of this work we design a reconfigurable cryptography processor core using three algorithms: (a) block cipher algorithms - Advanced Encryption Standard (AES) which supports 128 bits of data block and 128 bits of key size[6]; (b) RC5 which supports 64 bits of data block and 128 bits of key size [7], [9]; (c) RC6 which supports 128 bits of data block and 128 bits of key size [8], [10].

Our implementation is restricted to the following scope of work:

(a) The research is only to design fixed data block and Key sizes using only AES, RC5, and RC6 algorithms.

(b) The research is limited to design, to simulate, to verify the design correctness, to synthesize using Synopsys tool and to analyze the results.

(c) Use the test vectors based on FIPS 197, NIST (2002).

V. Process Flow

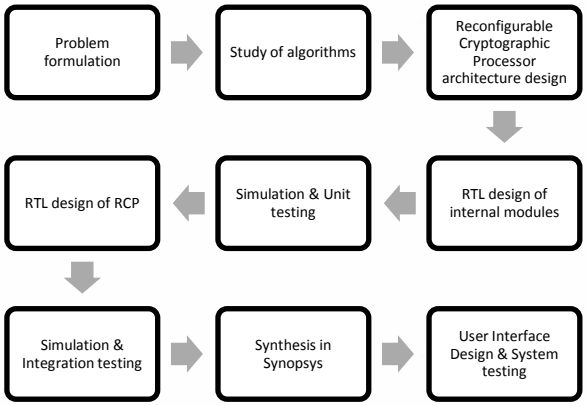

Figure 1. Process Flow

Fig 1 shows the entire implementation flow from problem definition, RTL design to synthesis and user interface design and system testing. Fig. 2 shows different methods by which cryptographic algorithms are implemented.

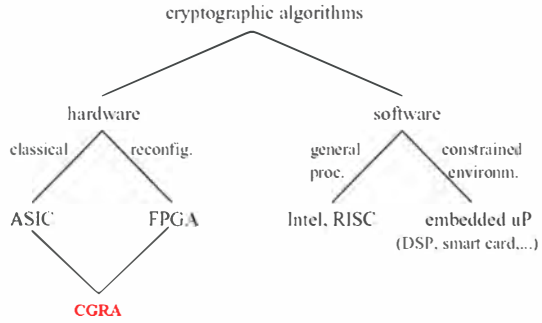

Figure 2. Implementation Platforms

Research Methodology, Techniques and Tools

The following figures 3 and 4 show the performance, flexibility and cost of various implementation platforms. From the figures we can conclude that Reconfigurable architectures are very suitable for implementing the cryptographic algorithms.

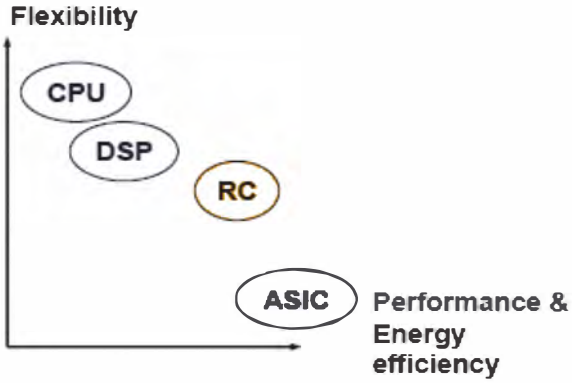

Figure 3. Performance Vs Flexibility Graph

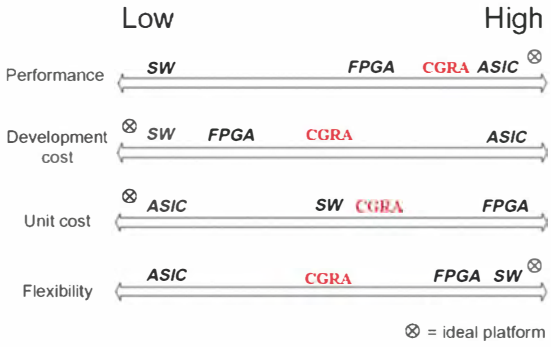

Figure 4. Parameters to evaluate implementation platforms.

VI. RECONFIGURABLE CRYPTOGRAPHIC PROCESSOR ARCHITECTURE

The proposed Reconfigurable Cryptographic Processor (RCP) architecture is similar to traditional micro-programmed Processor architecture. The Processor works only in the stand alone mode where the user has freedom to select an algorithm. Fig.5 shows the block diagram of RCP which contains four main blocks namely Reconfigurable Datapath, Data memory, Configuration memory, and Control Unit. The 32 bit reconfigurable datapath forms the core of Processor which can execute a set of cryptographic operations. The functionality of datapath can be reconfigured for performing various public key algorithms. Key features of the Processor which makes it distinct from previously implemented cryptographic Processor are,

- Architectural optimization schemes are explored to improve hardware utilization, and specifically resource sharing among different arithmetic algorithms is implemented to reduce the overall hardware requirement.
- The three different algorithms are implemented using same architecture.
- Other cryptographic block cipher algorithms can be implemented by just changing the set of configuration words.

Reconfigurable Datapath is made up of four Reconfigurable Cryptographic Blocks (RCB). Configuration memory is divided into four ROMs in which the configuration word for each RCB is stored. The data memory is divided into four RAMs which are used for temporary storage of round keys and intermediate values which can be later retrieved. Control Unit is made up of Configuration Controller and Data memory Address generator. It generates the control signal for read or write and address for data memory. The configuration controller is responsible for controlling datapath and for implementing cryptography algorithms by the issue of configuration words every cycle. It loads PC with next address which points to a configuration word to be given to each RCB. In the case of data independent operations, PC is incremented by one and in case of data dependent operations PC will be loaded with respected values depending on the algorithm and the data. With the Configuration word approach, cryptographic algorithms can be inserted or optimized without modifying the hardware implementation.

From the analysis of the algorithms, it was determined that a reconfigurable cryptographic processor core should support a 32-bit datapath replicated at least four times to allow for the implementation of algorithms requiring either 64-bit or 128-bit block lengths. The processor also supports communication between the 32-bit datapath to implement some of the operations in the algorithms in hardware.

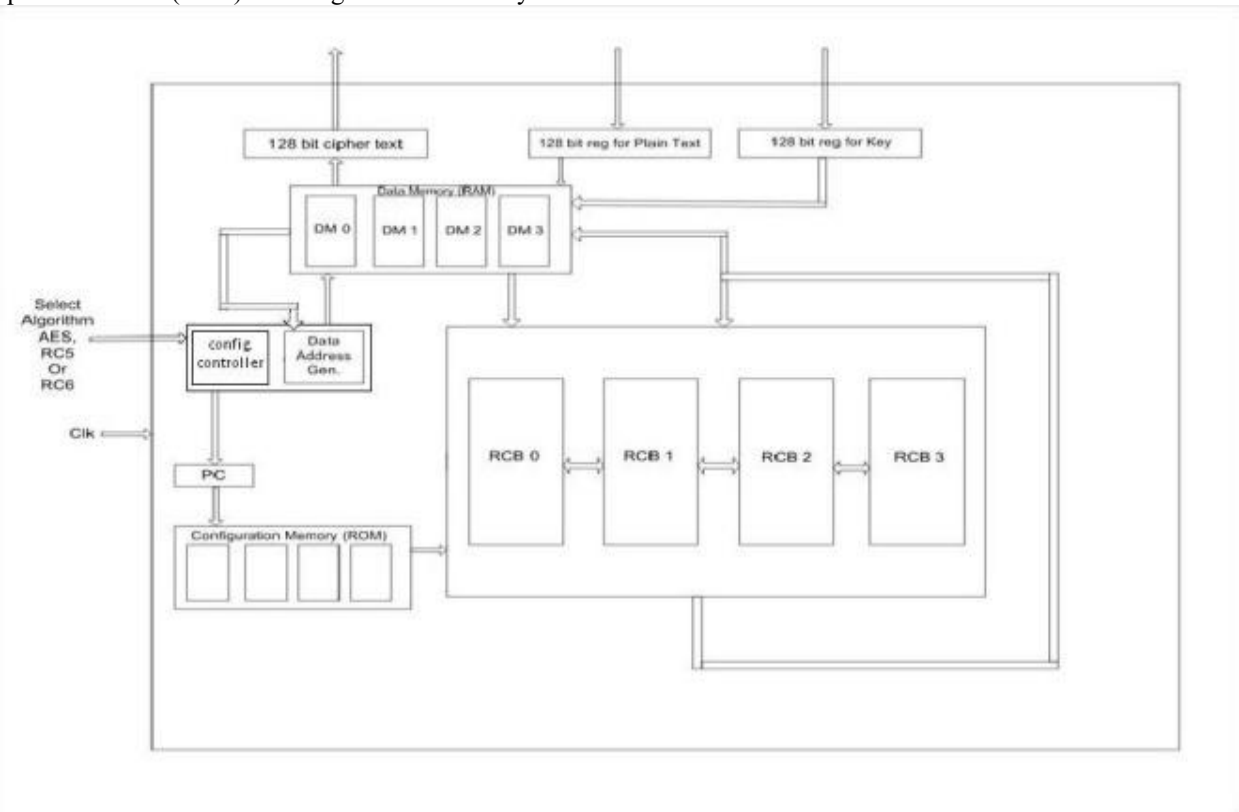

Figure 5. Reconfigurable Cryptographic Processor Block Diagram

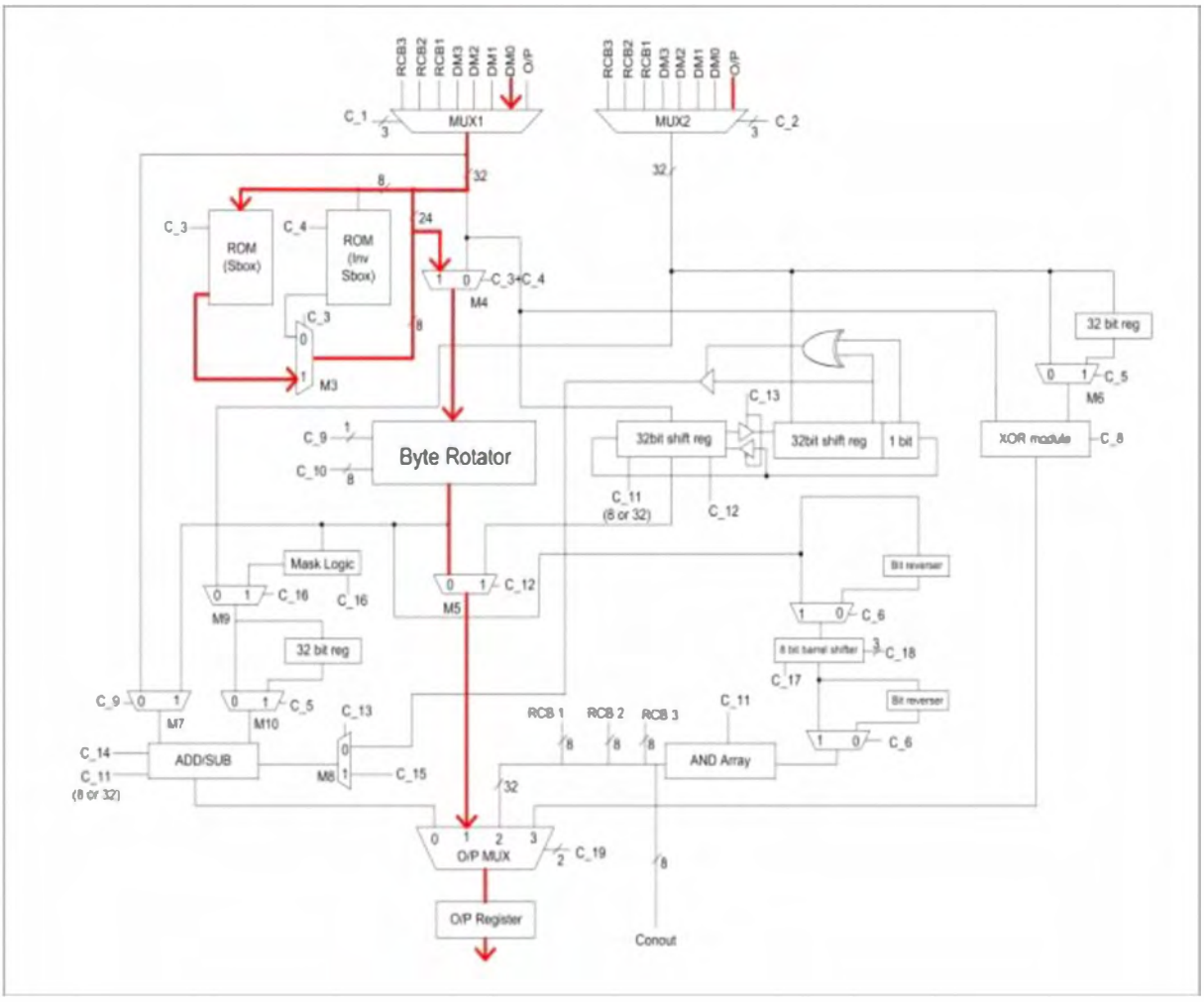

Figure 6. Reconfigurable Cryptographic Block

VII. RECONFIGURABLE CRYPTOGRAPHIC BLOCK

The RCB architecture is shown in Fig.6. The red color path indicates the sample flow of Sbox substitution for AES algorithm. It accepts 32 bit data and processes it according to the control signals given to it. The control signals are stored as configuration word and are given to RCB on each clock cycle. These signals are hard wired to different modules inside RCB and will allow for data to be processed. The inputs to RCB are from data memory, RCBs and output register and these are given to two 8 to 1 multiplexers MUX1 and MUX2. The various modules are Sbox ROM, Inverse Sbox ROM, Byte Rotator, Mask logic, Adder/Subtractor module, multiplier, 8 bit barrel shifter and XOR module.

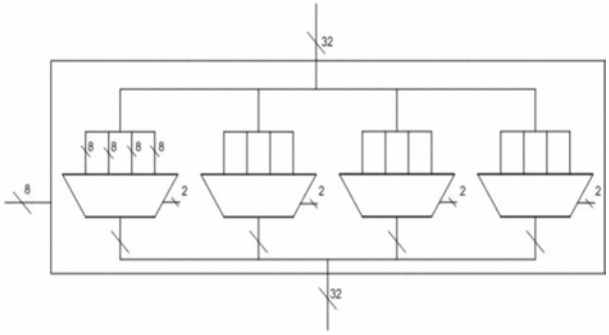

Figure 7. Byte Rotator Module

Byte rotator consists of four 4 to 1 multiplexers as shown in Fig 7. The inputs to the multiplexers are 8 bits wide and the outputs of all multiplexers are concatenated to form a 32 bit data output. Control signal for byte rotator is 8 bits and it is used to rotate the input by 0 or 8 or 16 or 24 times. If no rotation is needed then the control signal is 11100100 which mean the first multiplexer selects the

MSB 8 bits of the input and the last multiplexer selects the LSB 8 bits of the input thereby making no rotation.

A. MASK LOGIC

It is used to mask the input with 0s. It retains the LSB 8 bits and masks the other 24 bits with 0 and gives new 32 bit data which is masked. It is useful when there is a need to make an 8 bit operation including add/subtract. It keeps the 8 bit data to be operated and masks rest of the data which are redundant.

B. ADD/SUB MODULE

Control signal enables the module and either 8 or 32 bit operation can be performed depending on the signal for mode (8/32) of operation. C_14 is the enable for the module and C_15 gives the signal for add/sub operation. The module performs add if C_15 is 0 and perfoms subtraction if it is 1.

C. MULTIPLIER

This module can act as shift register and as multiplier. C_12 is the enable for the module and C_11 determines whether it is 8/32 bit operation. When C_13 is low, this module can rotate the 32 bit number by once each clock. If C_13 is high then it acts as booth multiplier. The last and penultimate bit is xored and given as enable for the buffer. If the bits are 00 or 11 then no action is taken. If the sequence is 01 or 10 then the buffer is enabled and the penultimate bit 0 or 1 is given as input to the add/sub module, which performs addition or subtraction depending on the input.

D. XOR MODULE

This module is used to xor two inputs to it and is enabled by the signal C_8. This module always performs 32 bit xor operation.

E. SBOX AND INV SBOX ROM

Sbox is implemented as a look up table which takes in 8 bit data and substitutes the data with another 8 bit data. The remaining 24 bit data is appended with the output to form a new 32 bit data. Inverse Sbox is implemented in the same way. As part of this work we design a reconfigurable cryptography processor core using three algorithms: (a) block cipher algorithms - Advanced Encryption Standard (AES) which supports 128 bits of data block and 128 bits of key size; (b) RC5 which supports 64 bits of data block and 128 bits of key size; (c) RC6 which supports 128 bits of data block and 128 bits of key size.

F. BARREL SHIFTER

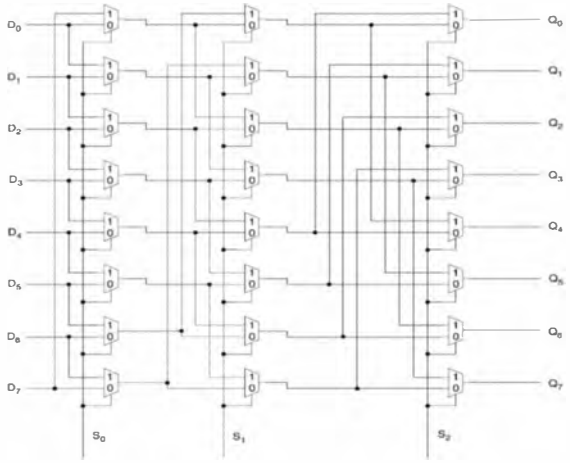

Figure 8. Barrel Shifter

Barrel shifter is used to rotate the input 8 bit number by a maximum of 8 times in a single clock. The internal construction of a barrel shifter is shown in Fig 8. It is an entirely combinational circuit made up of 24 multiplexers.

Table I: Various fields in configuration word

Field	Description	Field	Description
1	Selection line for MUX1	11	Select line for 8/32 bit operation
2	Selection line for MUX2	12	Enable for 32 bit Shift Register
3	Enable for S box	13	Enable for multiplier
4	Enable for inverse S box	14	Enable for add/sub module
5	Selection line for MUX6 & MUX 10	15	Add/sub control
6	Selection line for MUX to Reverse a byte.	16	Enable for mask logic
8	Enable for XOR module	17	Enable for barrel shifter
9	Enable for Byte rotator	18	Rotate amount
10	Selection line for byte rotator	19	Selection line for output MUX

VIII. CONFIGURATION WORD

The control signals to RCB are stored as configuration word in configuration memory. Table I shows how each configuration bit acts as control signal to the RCB.

IX. EXPERIMENTAL RESULTS, VALIDATION AND ANALYSIS

The Processor was designed using Verilog HDL, simulated using ModelSim and synthesized using Synopsys Design Compiler. The following are the results obtained using Synopsys.

A. Synopsys Results

ASIC chips need to be designed for particular applications; for different purposes or algorithms we need different chips. For a time varying application, which needs to switch application on time basis, more than one ASIC chip need to be designed and implemented. So the area to which the design maps will surely be more. In the reconfigurable system which we designed takes less area than this. In RCP same blocks can be reused for different applications. That is why the area is reduced by around 25% of that of ASICs. Fig 9 shows the bar diagram with area on y axis. Same algorithms designed as ASIC and as reconfigurable architectures are compared using synopsys design compiler and plotted. ASIC takes more area (94030 micro meters) than RCP (71663 micro meters). The configuration memory which is nothing but program memory can be kept outside of RCP and thus the area required for configuration memory can be reduced. Area required for RCP without the configuration memory (69820 micro meters) is also plotted along with the other two. Our design takes very less area compared ASIC design.

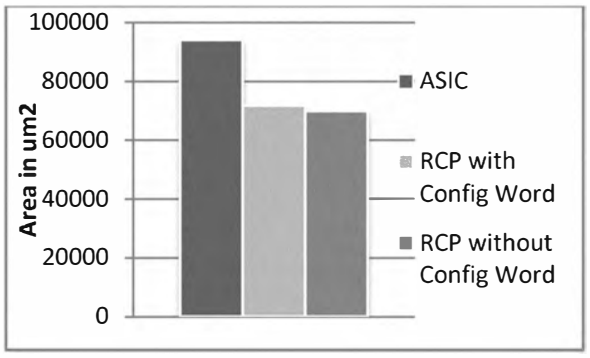

Figure 9. Area Analysis

Power needed to operate a circuit is proportional to the frequency of operation and the area requirement of the logic. We have compared here, two types of architectures. One is ASIC, which requires more area to map the logic, consumes more power. Reconfigurable architecture requires less area so the power consumed will also be less. One more reason for lower power consumption for reconfigurable architecture is due to disabling the modules when not in use with the help of configuration words. Fig 10 shows a plot of power consumption among the three types of architecture. We can see from the chart that the power consumed by the ASIC logic (8.9562mW) is around 28% more than that of RCP architecture with configuration word (6.555 mW) and around 37% more than that of RCP without configuration word(5.6003mW). A reconfigurable architecture is chosen if the objective is area and power efficiency and flexibility.

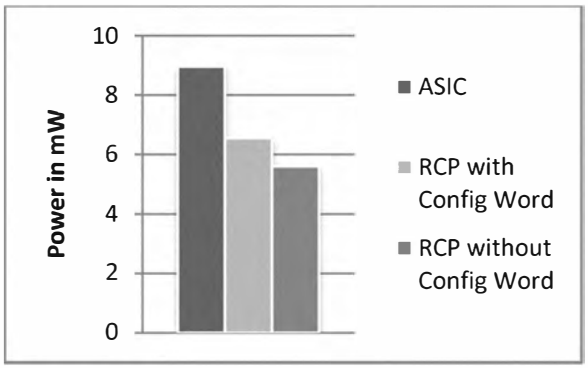

Figure 10. Power Analysis

Fig 11 shows the comparison among the three designs on critical path delay in nanoseconds. Implementation in ASIC is very faster compared to our reconfigurable design.

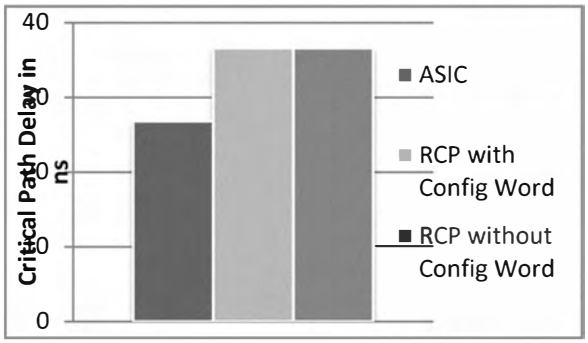

Figure 11. Critical Path Delay

A Graphical user interface was designed using java to input the plain text and key and to show the output to the user. The user can select one of the algorithms provided to encrypt/decrypt data. Whenever inputs are given and encrypt/decrypt button is pressed ModelSim is triggered and a file is created which will be read by ModelSim. Once the operations are over the output is written to the

text file which in turn is read by the Graphical user interface.

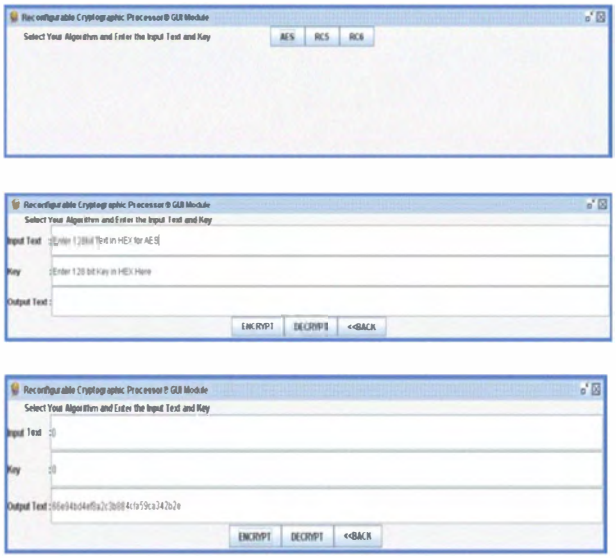

Figure 12. Graphical User Interface

X. CONCLUSION

The objectives specified in the introduction of this work have been achieved. A Reconfigurable Cryptographic Processor has been designed in Verilog HDL, simulated in Modelsim and synthesized in Synopsys. Both encryption and decryption operations were carried out in the scheme and the results validate the functionality and practicality of the proposed processor. The most significant aspect of this novel processor is that it is very flexible in the sense that three algorithms can be used to encrypt data using the same architecture and as well as be power and area efficient. Security wise the processor offers a provision to encrypt data by mixing the algorithms, allowing the users to fine tune an algorithm or make their own algorithms.

XI. ACKNOWLEGEMENT

We gratefully acknowledge the Almighty GOD who gave us strength and health to successfully complete this venture. The authors wish to thank the university management for access to digital library and research facilities.

REFERENCES

[1]. Byeong Kil Lee, Lizy Kurian John, "Implications of Executing Compression and encryption Applications on General Purpose Processors", *IEEE Transactions On Computers*, Vol. 54, No. 7, July 2005

[2]. Tsuyoshi Hamada, Khaled Benkrid, Keigo Nitadori and Makoto Taiji, "A comparative study on ASIC, FPGAs, GPUs and general purpose processors in the O(N2) gravitational N-body simulation", *NASA/ESA Conference on Adaptive Hardware and Systems*, 2009

[3]. Mancia Anguita, Javier D´ıaz, Eduardo Ros, and F. Javier Fern´andez-Baldomero, "Optimization Strategies for High-Performance Computing of Optical-Flow in General-Purpose Processors", *IEEE Transactions On Circuits And Systems For Video Technology*, Vol. 19, No. 10, October 2009

[4]. Dejan Markovic, Borivoje Nikolic, and Robert W. Brodersen, "Power and Area Minimization for Multidimensional Signal Processing", *IEEE Journal Of Solid-State Circuits*, Vol. 42, No. 4, April 2007

[5]. M. Ernst, **S.** Klupsch, 0. Hauck, and **S. A.** Huss, "Rapid Prototyping for Hardware Accelerated Elliptic Curve Public-Key Cryptosystems", 1074-600901, 2001

[6]. Joan Daemen and Vincent Rijmen, The Design of Rijndael: AES - The Advanced Encryption Standard (Information Security and Cryptography), *Springer Series*.

[7]. AES secured Hardware Cryptography lecture series from Virginia Tech.

[8]. Hua Li, Jianzhou Li and Jing Yang, "An efficient and reconfigurable architecture for RC5" *Conference on Electrical and Computer Engineering*, 2005, Page(s): 1648 - 1651

[9]. May Itani, and Hassan Diab, "Reconfigurable Computing For RC6 Cryptography", *Proceedings of the IEEE/ACS International Conference on Pervasive Services (ICPS'04)*

[10]. Ronald L. Rivest, "The RC5 Encryption Algorithm", *Proceedings of the 1994 Leuven Workshop on Fast Software Encryption (Springer 1995), pages 86-96*..

[11]. Ronald L. Rivest, M.J.B. Robshaw, R. Sidney, and Y.L. Yin, "The RC6 Block Cipher", Posted at *RSA's RC6 page*, (Version 1.1; August 20, 1998).

[12]. Symth, N.; McLoone, M.; McCanny, J.V., " Reconfigurable Processor for Public-Key Cryptography", *IEEE Workshop on Signal Processing Systems Design and Implementation, 2005.*

[13]. Garcia, A., Berekovic, M., Aa, T.V., "Mapping of the AES cryptographic algorithm on a Coarse-Grain reconfigurable array processor", *International Conference on Application-Specific Systems, Architectures and Processors*, 2008. ASAP 2008.Page(s): 245 - 250