“Chakravyuh” - The Digital Fortress

Dola Ram[1], Senthil V[2], Sirish K[3] and Dr. Chetan Singh Thakur[4]
Department of Electronic Systems Engineering,
Indian Institute Of Science, Bangalore, Karnataka, India - 560012
Contact: dolaram@iisc.ac.in

***Abstract*— This paper presents “Chakravyuh”, a simple, bare-bones security microprocessor to demonstrate the ideas of authentication, asymmetric-key cryptography using the celebrated RSA algorithm as well as secure on-ship storage of persistent data. The motivation for designing this security processor was to attempt the design and implementation of an in-house embedded RoT (Root-Of-Trust) which could be used in India on commercial security systems like CAS (Conditional Access Systems) used in broadcast television applications.**

I. INTRODUCTION

Security systems deployed in various commercial applications like Conditional Access Systems (CAS) in Broadcast TV require an embedded RoT (Root-Of-Trust) which provides cryptographic authentication as well as encryption / decryption capabilities to ensure that the system is “hacker-proof”.

Such RoT chips typically consist of a microprocessor to perform cryptographic operations and a tiny embedded memory unit to store data in encrypted format (like DRM keys & user entitlements in Broadcast TV applications).

In this research study, an attempt was made to design and implement a simple security processor which could be integrated into a larger embedded security system (like the commercial CAS found in Broadcast TV) with the following key features :

I) Authentication of CPU prior to command processing

II) Support for the following operations post authentication :

- Encryption of input data using RSA algorithm
- Decryption of input data using RSA algorithm
- Secure storage & retrieval of an input data word at a particular address in the embedded memory

III) Novel Vedic multiplier & Enhanced encoder based multiplier units borrowed from research ideas developed in the past (cited in the “Bibliography” section for reference)

The motivation for this study was to simply explore some ideas in security, cryptography and digital systems design as an integral part of a mini-project in the course “E0-284: Digital VLSI Circuits” offered by the Department of Electronic Systems Engineering, IISc Bangalore.

The makers-in-mischief claim no novelty of the ideas presented, this was simply a means to do some interesting systems design work using HDL & have fun in the whole process !

II. SOME BACKGROUND IDEAS

Let us begin at the beginning, and explain some important ideas used in this study before moving onto the design and implementation aspects of the same.

A. RSA asymmetric key cryptography algorithm

RSA is an asymmetric key cryptography algorithm which derives its security from the difficulty of factoring large integers that are the product of two large prime numbers.

Multiplying these two numbers is easy, but determining the original prime numbers from the total i.e. factoring; is considered infeasible due to the time it would take even using today’s super computers.

Here’s an example to illustrate its principle of operation and to see how it works :

1. The sender of the information, say “Alice” generates 2 primes p=11 and q=13.

2. A “modules” is calculated based on these 2 numbers as :

$$n=p\times q=143$$

3. Next, a “totient” is calculated as follows :

$$\phi(n)=(p-1)x(q-1)=120$$

4. Alice chooses 7 as her RSA public key *e* and then calculates her private key *d* using Extended Euclidean Algorithm such that the following condition is satisfied :

$$e\cdot d=1 \bmod \phi(n)$$

5. This gives *d* as 103.

6. Next, Bob wishes to send Alice an encrypted message *M. S*o he obtains her RSA public key (n, e) which in this example is (143, 7). His plaintext message is just the number 9 and is encrypted into ciphertext C as follows:

$$M^e \bmod n = 9^7 \bmod 143 = 48 = C$$

7. When Alice receives Bob’s message she decrypts it by using her RSA private key (d, n) as follows:

$$Cd \bmod n = 48^{103} \bmod 143 = 9 = M$$

The above explanation is borrowed from the website “*http://searchsecurity.techtarget.com/definition/RSA*”.

B. Authentication

Authentication is the process of proving one's identity to someone else. As humans, we authenticate each other in many ways: we recognize each other’s faces when we meet, we recognize each other’s voices on the telephone, we are authenticated by the customs official who checks us against the picture on our passport.

Now why are we interested in authentication in the field of electronics ?

The answer is simple : There are certain applications in the world which require exchange of data which is confidential and which needs to be securely communicated between a set of participants. When this data is communicated across a certain channel (it could be air / wire); the data transmitted becomes susceptible to eavesdroppers who might be interested in the information being sent.

Furthermore, these eavesdroppers might masquerade themselves as the intended recipient of the communicated data and may misuse the same for their nefarious schemes !

So the idea here is to ensure that prior to communication of data between 2 parties, each party verifies each other’s identity; thereby establishing a link of trust before sharing any information across the channel.

There were many authentication protocols devised in the past, and currently the most popular protocol is the one depicted in the below diagram :

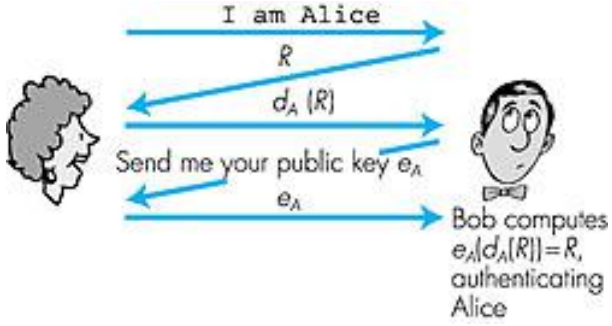

Fig. 1 : Authentication using a nonce & asymmetric key encryption with RSA (image courtesy : http://netlab.ulusofona.pt/rc/book/)

Here’s what happens :

1. 2 parties Alice & Bob wish to communicate with each other.

2. Alice has to prove her identity to Bob.

3. Alice says “I am Alice” to Bob.

4. Bob challenges Alice and asks her to encrypt a random number (a.k.a. “nonce”) with her private key.

5. Alice encrypts the same and sends it back to Bob.

6. Bob uses Alice’s public key to decrypt the encrypted nonce, compares the decrypted value against the original nonce sent and it they match; he could conclude with some

degree of certainty that it is indeed Alice who is communicating with him.

Though this approach works well, there's a pitfall which is often known as the MITM / Man-In-The-Middle attack :

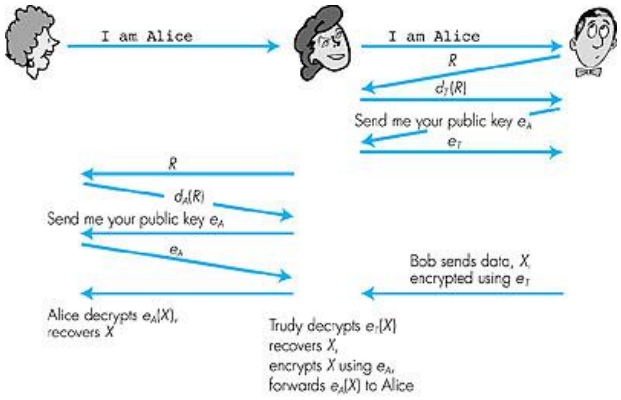

Fig. 2 : Man-In-The-Middle attack (image courtesy : http://netlab.ulusofona.pt/rc/book/)

Trudy, a malicious hacker; could easily impersonate herself as Alice to Bob and as Bob to Alice and compromise the security of the communication channel.

A possible solution to the above problem is the use of Digital Certificates along with an escrow party which manages certificates for all registered parties. The discussion of this is beyond the scope of this paper.

C. Vedic and encoder based multiplier units

The RSA cryptographic encryption / decryption operations rely heavily on the exponentiation function; which requires an efficient multiplier unit to carry out the same.

For this, the authors discovered an interesting Vedic multiplier unit as well as an encoder based multiplier unit developed from research carried out in the past (see the "Reference" section for the full list of papers regarding the same).

The motivation was simply to try implementing them, and perhaps compare their performance and see which one was better.

i) The Vedic multiplier :

Urdhva Triyagbhyam is a multiplication Sutra (Algorithm) in Vedic Mathematics. This is the general formula applicable to all cases of multiplication. Urdhva means Vertical and Triyagbhyam means Crosswise.

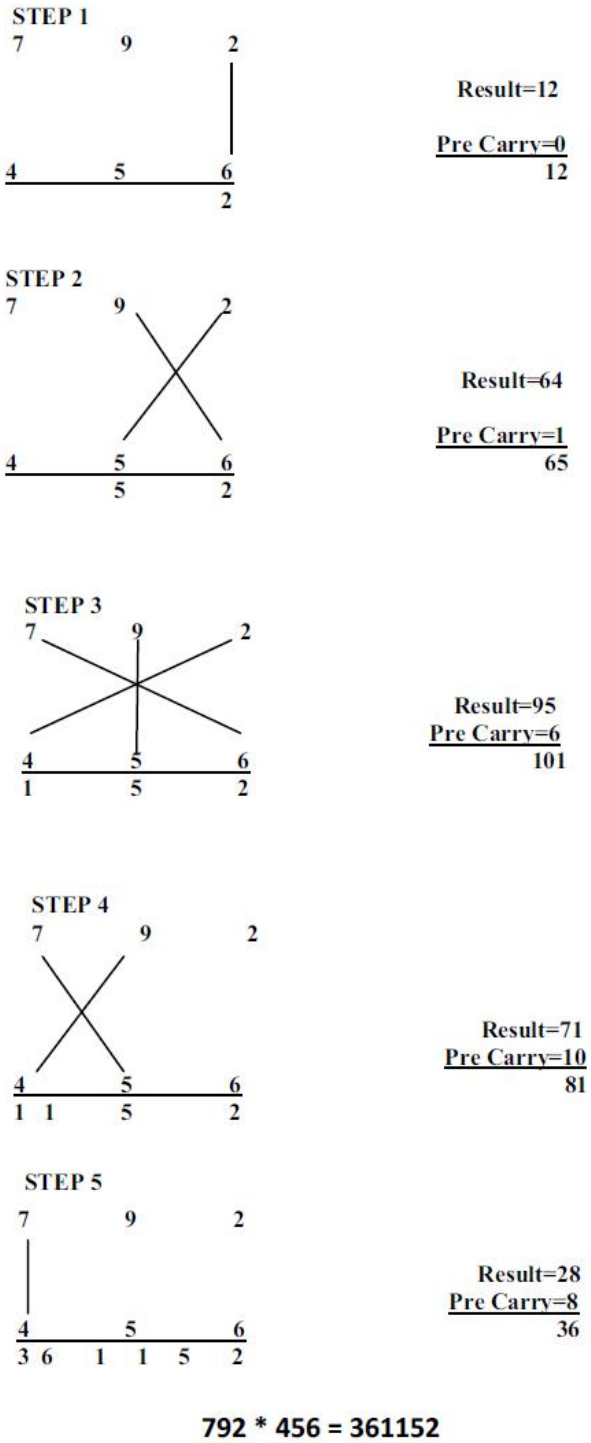

Fig. 3 : Diagram depicting how 2 numbers are multiplied using the Vedic multiplication approach (image courtesy : "Design and Analysis of ALU: Vedic Mathematics Approach" by Garima Rawat & Co.)

The above figure depicts how multiplication is accomplished for two 3 digit decimal numbers. The same approach is followed even for numbers represented in binary.

For designing the Vedic multiplier in hardware, a basic 2x2 multiplier is realized, which is used to develop a 4x4 multiplier which is again used to develop an 8x8 multiplier and so on.

Hence, a nice recursive structure is obtained as follows :

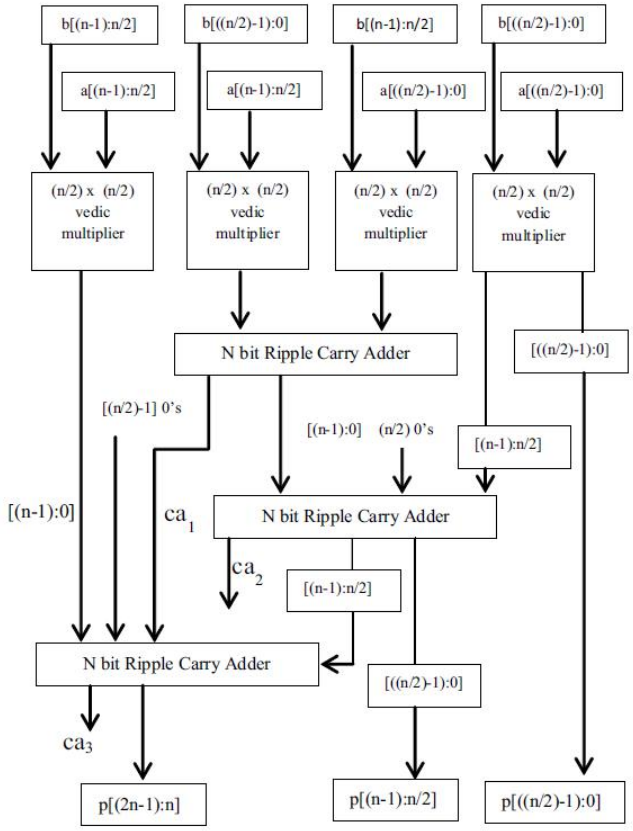

Fig. 4 : Diagram depicting how 2 numbers are multiplied using the Vedic multiplication approach (image courtesy : "Design and Analysis of ALU: Vedic Mathematics Approach" by Garima Rawat & Co.)

ii) Encoder based multiplier :

The encoder based multiplier is another idea which cleverly reduces the huge number of multiplication operations involved in the Vedic multiplier as well as number of gates required to realize the operation.

Here, the complexity of the multiplication process is reduced by breaking the multiplier (which multiplies the multiplicand) into groups of 2 bits (say [bi+1,bi]) and performing the following operations based on their value; here's the pseudo code :

```
pp (i) = 0;

If [bi+1,bi] equals 1
pp (i) = multiplicand;

Else If [bi+1,bi] equals 2
pp (i) = shift the multiplicand 1
         position left;


Else If [bi+1,bi] equals 3
pp (i) = sum of pp (i) for code 1 & 2.
```

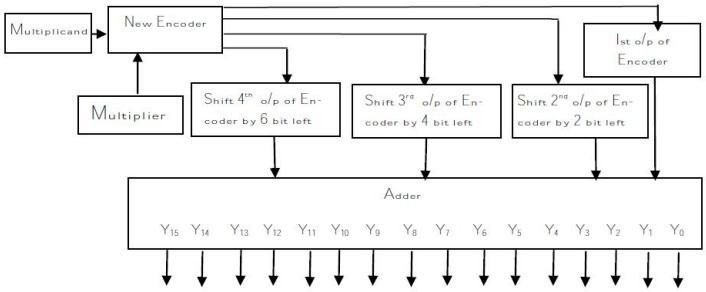

Fig. 5 : Diagram depicting how 2 numbers are multiplied using the encoder based multiplier (image courtesy : "An Efficient Design of Vedic Multiplier using New Encoding Scheme" by Jao Skand Tripathi & Co.)

III. HIGH LEVEL DESIGN

The system was designed using a top-down approach, the below diagram depicts the module-wise breakup of the whole system :

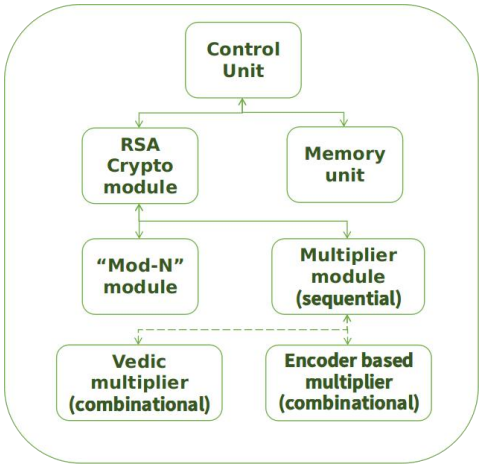

Fig. 6 : Diagram module-wise breakup of "Chakravyuh"

The functionality of each module is as below :

- *Control Unit* : To act as the user interface to the CPU, orchestrate authentication procedure as well providing opcode support to perform RSA encryption / decryption and storage / retrieval of data to the on-chip memory.

- *RSA Crypto Unit* : To perform the mathematical operation "e^x *mod n*" by using the multiplier as well as mod-n modules.

- *Mod-n Unit* : To perform "*x mod n*" mathematical operation by performing repeated subtraction of *n* from *x* till x becomes less than *n*.

- *Vedic multiplier & Encoder based multiplier units* : Combinational logic which carries out multiplication of two 8-bit binary numbers and gives a 16-bit result.

- *Sequential multiplier unit* : This module acts as a wrapper interface between the combinational multiplier unit(s) and the rest of the system whose logic is sequential in nature.

- *Memory* : This unit provides space to store 256 words of 8 bits word size. It provides the service to store / retrieve data bytes to / from a memory location addressed using 8 bits.

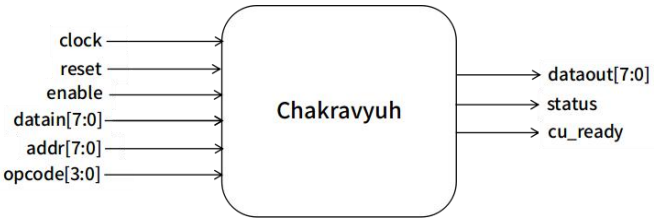

Fig. 7 : Black-box view of "Chakravyuh" with the lines on the left-hand side indicating input lines & the lines on the right-hand side indicating output lines

IV. CONTROL & DATA FLOW

The following diagram depicts how the modules interact with each other in a typical sequence of operations involving authentication followed by storage of a data word by the CPU :

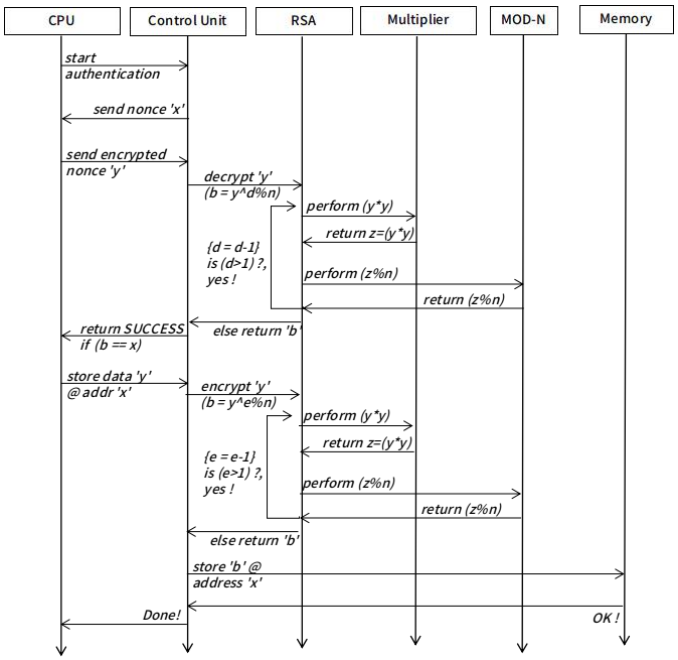

Fig. 8 : Control & Data flow diagram depicting a typical use-case of the chip where the CPU wishes to store data onto Chakravyuh's on-chip memory

At first, the CPU has to perform a reset of Chakravyuh by asserting the "reset" line for a clock cycle and waiting till "cu_ready" signal goes low (logic "0").

Following this, the CPU has to authenticate itself, otherwise Chakravyuh would not allow him to perform any of the supported operations. This is done by setting the appropriate opcode for the same and then setting the "enable" line to high.

Once the authentication opcode is set, Chakravyuh would send a challenge "nonce" back to the CPU which the CPU would have to encrypt and return.

Following this, control unit would receive the encrypted nonce from the CPU, exercise the RSA module to decrypt the same with the public key of the host CPU.

The RSA module, in turn would invoke the multiplier and mod-n units to perform "e^x mod n" operation and once it's done, return the result back to the control unit.

Finally, the control unit would check if the decrypted value matches the original nonce sent and if they match, a success status code is returned back to the CPU.

Thereafter, the CPU is allowed to perform any of the below supported operations as previously mentioned in the introductory section :

- Encryption of input data using RSA algorithm
- Decryption of input data using RSA algorithm
- Secure storage & retrieval of an input data word at a particular address in the embedded memory

V. Implementation & Results

The below diagrams show the synthesis of the RTL code written to describe the entire system :

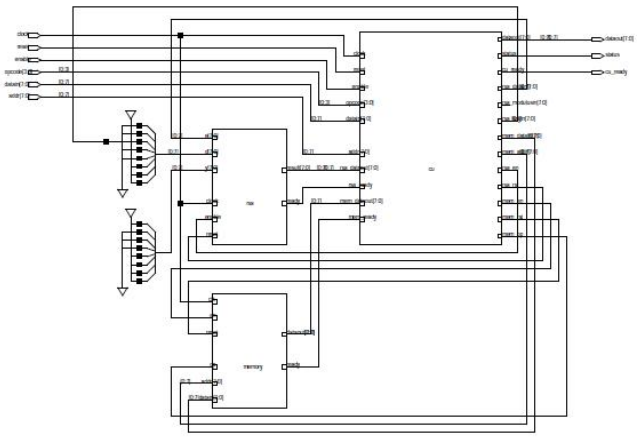

Fig. 9 : Schematic view of entire system; Top-Left : RSA module, Bottom-Left : Memory module, Top-Right : Control Unit

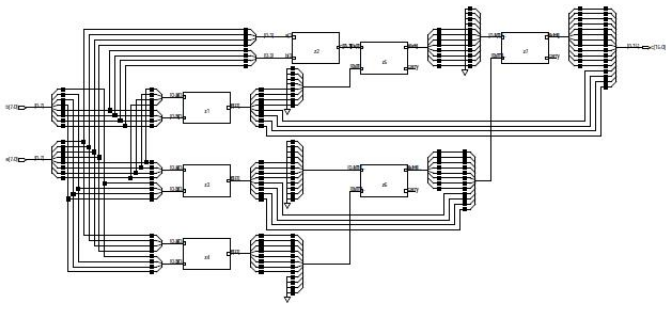

Fig. 10 : Schematic view of the (8bit X 8bit) Vedic Multiplier Unit

Since the images of the rest of the modules are either too big or complex to fit into the limited space of this report, they have been omitted from display.

The following timing diagram captured using “IRun” simulation tool provided by Cadence depicts the system in action. Here, the testbench first performs authentication followed by 3 memory write, operations, 3 memory read operations and finally 3 RSA encryption and decryption operations back-to-back.

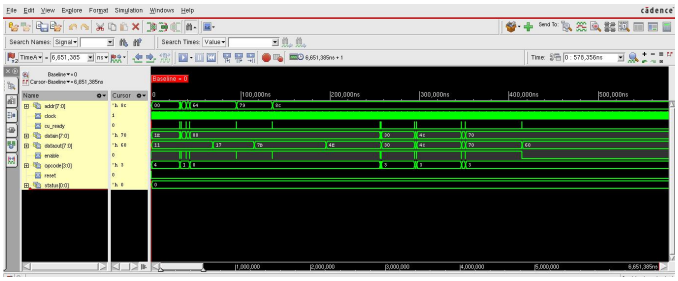

Fig. 11 : Timing diagram showing the chip in action

All operations performed were verified to be logically correct and the system works as per the intended objectives.

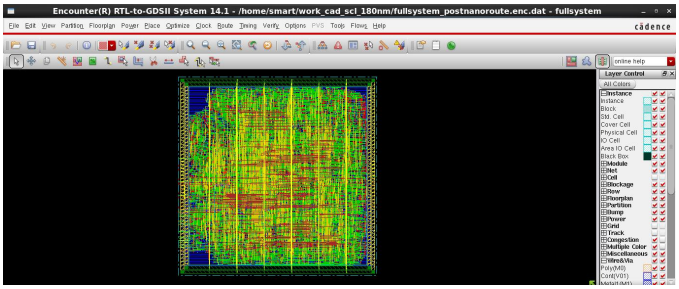

Fig. 12 : Layout view of the whole system Cadence’s Encounter tool

The above figure captures a snapshot view of the final layout of the chip.

VI. Conclusion & Scope For The Future

A fully functional albeit simple security processor was realized to support the functions listed in the introductory section.

The system is secure as along as generation of the nonce by the security chip during the authentication stage is guaranteed to be truly random. Practically, this could be realized by having a separate battery powered random number generator chip which generates a constant stream of random numbers per clock cycle.

There is scope for further enhancements as envisioned below :

- Diffie-Hellman key exchange post authentication stage followed by encryption of all communications (information flowing through the opcode, datain / dataout and address lines) between the CPU and security chip.
- Data and control line obfuscation / mixing for added security against hacker attacks.
- Intruder detection system to raise an alarm if the number of invalid authentication failures.
- Random number generator module for generating the nonce on-the-fly. For now, the system has a hard-coded nonce.
- The system is still susceptible to MITM attacks. This could be solved by implementing Digital Certificates.

ACKNOWLEDGMENT

We would like to thank Dr. Chetan Singh Thakur for giving us the opportunity to work on this idea which we'd envisioned as a fun mini-project for the course "E0-284 Digital VLSI Circuits".

We would also like to thank Mr. Kundan Kumar for helping us setup the development environment for this project.

REFERENCES

[1] Jai Skand Tripathi, Priya Keerti Tripathi, & Deepti Shakti Tripathi, *An Efficient Design of Vedic Multiplier using New Encoding Scheme, International Journal of Computer Applications (0975 – 8887) Volume 53– No.11, September 2012*

[2] Jinyoung Moon, Jongyoul Park, and Euihyun Paik, *JavaCard-based Two-Level User Key Management* , Home Network Group, Digital Home Research Division, Electronics and Telecommunication Research Institute, 161 Gajeong-dong, Yuseong-gu, Daejeon, Korea.

[3] Dr. K.S. Gurumurthy, M.S Prahalad, *Fast and Power Efficient 16×16 Array of Array Multiplier using Vedic Multiplication*, DOS in Electronics and Communication Engineering, UVCE Bangalore

[4] Dani George, Bonifus P.L., *RSA Encryption System Using Encoded Multiplier and Vedic Mathematics,* 2013 International Conference on Advanced Computing and Communication Systems (ICACCS -2013), Dec. 19 – 21, 2013, Coimbatore, INDIA

[5] Garima Rawat, Khyati Rathore, Siddharth Goyal, Shefali Kala and Poornima Mittal, *Design and Analysis of ALU: Vedic Mathematics Approach,* International Conference on Computing, Communication and Automation (ICCCA2015)

[6] The Search Security Website : http://searchsecurity.techtarget.com/definition/RSA

[7] *The book "Computer Networks : A Top Down approach with Networking" : http://netlab.ulusofona.pt/rc/book/7-security/7_03/index.htm*