A

DISSERTATION REPORT

On

FPGA based Secure and Reliable Face Recognition System

By

Vipul Sharma

2015PEV5186

Under the supervision of

Dr. Amit M Joshi

Assistant professor, ECE Department

MNIT, Jaipur

Submitted in partial fulfillment of requirement of degree of

MASTER OF TECHNOLOGY

To the

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

JULY – 2017

Department of Electronics and Communication Engineering

Malaviya National Institute of Technology, Jaipur

Certificate

This is to certify that this Dissertation report entitled ***"FPGA based Secure and Reliable Face Recognition System"*** by **Vipul Sharma** (2015PEV5186), is the work completed under my supervision and guidance, hence approved for submission in partial fulfillment for the award of degree of ***Master Of Technology*** in ***VLSI Design*** to the Department of Electronics and Communication Engineering, Malaviya National Institute of Technology, Jaipur in the academic session 2016-2017 for full time post-graduation program of 2016-2017. The contents of this dissertation work, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Amit M Joshi

Assistant Professor, ECE Department

MNIT, Jaipur

Declaration

I, hereby declare that the work which is being presented in this project entitled "FPGA based Secure and Reliable Face Recognition System " in partial fulfilment of degree of Master of Technology in VLSI is an authentic record of my own work carried out under the supervision and guidance of Dr.Amit M Joshi in Department of Electronics and Communication, Malaviya National Institute of Technology, Jaipur.

I am fully responsible for the matter embodied in this project in case of any discrepancy found in the project and the project has not been submitted for the award of any other degree. I also confirm that I have consulted the published work of others, the source is clearly attributed and I have acknowledged all main sources of help.

(Vipul Sharma)

Acknowledgement

I am grateful to my supervisor Assistant Professor Dr. Amit M Joshi for his constant guidance and encouragement and support to carry out this work. His excellent cooperation and suggestion provided me with an impetus to work and made the completion of work possible. He has been great source of inspiration to me, all through. I am very grateful to him for guiding me how to conduct research and how to clearly & effectively present the work done.

I would like to express my deepest sense of gratitude and humble regards to our Head of Department Prof. K.K Sharma for providing necessary facility in the Department. I am very thankful to Mr. Bharat Kashyap, Lab Assistant and all other faculty members of ECE, MNIT for their valuable assistance and advice.

I would also like to thank my friends for their support in discussions which proved valuable for me. I am indebted to my parents and family for their constant support, love, encouragement, and sacrifices made by them so that I could grow up in a learning environment.

Finally, I express my sincere thanks to all those who helped me directly or indirectly to complete this work successfully.

(Vipul Sharma)

Abstract

With the advent of technology every day and night people are working to make human life easier so that the lesser amount of time is spent in doing trivial things and people can focus more on important things. Biometric recognition is being used in various applications ranging from gadget personalization, security, surveillance systems etc. Face is considered best way to authenticate among all the biometric authentication techniques as it has better FAR (False Acceptance Ratio), FRR (False Rejection Ratio), error is minimized as there is no physical contact is needed with the system. Securing the biometric information is important as it is the personal information of the user which could be exploited. The idea is to design and implement such a system on FPGA (Field Programmable Gate Array) that will provide a secure face recognition system in which biometric data is not saved in the database directly instead, random data is saved along with a hash function which is matched during authentication.

Second part of this thesis is dedicated to the work carried out at the Fermedicius Pvt. Ltd., Bangalore. Being a part of the Internet of Things and System Engineering, the work is focused on designing and developing a smart base that provides a personalized assistance to the users about the food product that they have in their house.

Contents

List of Figures

Chapter 1

Introduction

Today in this technologically driven society, there are many security and privacy related issues being faced every day and reliable user authentication is one of them. Traditional password based authentication system may be considered secure enough but the level of security these systems provide is limited to relatively weak human memory and ease of access to the authentication system. So, this is not the preferred method for system which require high level of security. More reliable approach is to use biometric (fingerprint, iris, face, voice) instead of the password for authentication. High entropy and uniqueness make them favourable in so many applications. Biometric recognition has great potential in authentication of a valid user and allow access to a particular securely kept document, place etc.

It has two phases, initially the biometric data of a user is enrolled in the form of a template. Usually a set of multiple images are taken to enhance the tolerance in the variation during authentication phase when user present himself/herself in front of the image sensor to authenticate. Even biometric templates have some shortcomings are addressed in the following text and an attempt to provide a reliable solution has been made. Face recognition is focused in this thesis work as it has better FAR, FRR values and subject distance can be longer as compared to other biometric recognition techniques.

1.1 Motivation

Biometric authentication techniques were introduced as the process of manual password protection of documents had a lot of overhead involved in remembering the long passwords. A lot of work has been done in developing more and more efficient recognition systems over the time. But protecting those biometrics is equally important as well as they have person information of the users like fingerprint, face, eye, voice etc. and these data can be used to exploit the users. Not much work has been done in protecting these biometrics. First part of this these is dedicated toward designing and implement a secure biometric system Verilog hardware descriptor language. Face biometric provides the lowest FRR and FAR amongst all biometric techniques. SHA-2(256) is used in securing data indirectly related to the biometric information.

So, this recognition system provides a system which reliable and secure at the time.

In the second part of this thesis, a smart device is designed and developed which focuses on the health and wellbeing of the users by assisting them about the food products they consume. People in this fast paced life, give very less priority to their health. Developed device provides assistance to the user by keeping the track of their daily consumptions and alerting them when to eat and what to eat.

1.2 Problem statement

During authentication, the conditions may vary for example variation in the background, changes in human face, angle for face , expression, illumination etc. all these conditions can have an impact on proper authentication. There are various algorithms specifically for face recognition that have addressed this issue and tried to resolve them up to some extent. Following points provide the challenges that are hampering the widespread use of the face biometric authentication systems:

- Variation between the face template stored during enrolment phase and the template presented during authentication phase. There are many variations as discussed above can cause reduce the efficiency of any face recognitions.
- Storage space required to store the face data. As the size of the images for face recognition ranges from 50x50 to 500x500. Huge amount of space is required to store multiple images of multiple users.
- Speed of the authentication process. Since there are multiple computations that take place in authentication phase, huge amount of data needs to be processed in very little amount of time in order to make the recognition process efficient and practical.
- Security of the biometric templates. This issue has not been addressed the way face recognition is. These templates contain the biometric information that once revealed would allow an attacker to obtain sufficient information to impersonate the original user. This is a challenging issue to address as in this age of computers with processing speed going beyond 4Gbps, it is very difficult to build a system that can never be compromised. Figure1 shows all the possible attacks that can occur on a biometric authentication system.

The compromised template cannot be revoked like passwords. So, this problem is difficult as compared to the traditional authentication systems based on password or certificates. Since biometric templates are very much prone to the variations discussed above, direct use of cryptographic hash functions would not be the ideal choice to hide the biometric template the way passwords are hidden.

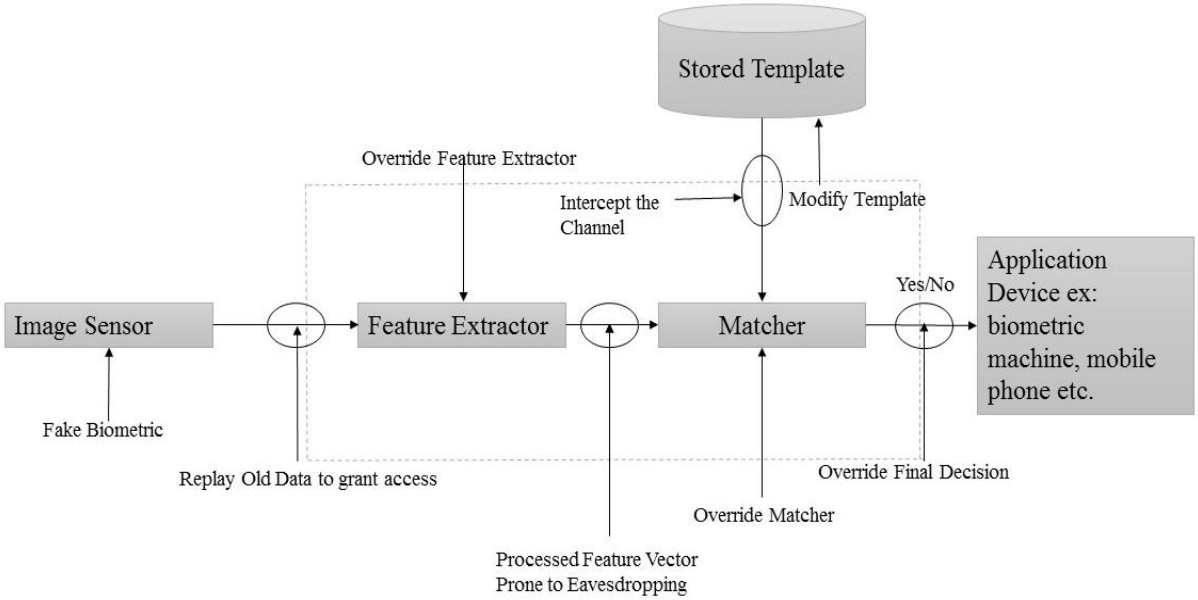

Figure 1 Possible Attacks on an Authentication System

Fuzzy commitment provides the basis in which the fuzziness of the biometric data is addressed and methods are used to compute that variation in the biometric and correct them to the originally enrolled biometrics. Reed-Solomon has been used to correct those variations. Reed-Solomon does not directly use biometric data as the input corrupted data to the Reed-Solomon decoder but some randomly data generated using pseudo random generator. Hashing is used in such a way that it does not operate on the input biometric template.

Presented work provides an end to end solution which during authentication phase, consider the input biometric data as a corrupted template. After that Reed-Solomon cods along with hashing are used to correct and securely store and match the data in order to authenticate the user. FPGA implementation of the whole system provides the concurrency between independent modules which enhances speed of the system.

1.3 Objective

Objective of this thesis is to design and implement an end-end face recognition system that keeps the biometric data related information securely in the database so that eavesdropping can be avoided. In order to achieve this goal, following prerequisites are to be fulfilled:

- Design of a fuzzy commitment algorithm tailored for face recognition system.
- Design of Secure Hash Function-2 with digest value of 256-bit.
- Design of an LFSR (Linear Feedback Shift Register) of 223-bytes (1784-bits).
- Design of a Reed Solomon encoder/Decoder with 255-byte code word and 223-byte message.

1.4 Thesis outline

In Chapter 2, Literature of the related work that has been done and background of the secure biometric authentication system is given. In chapter 3, Proposed work is described in details and various aspect addressed in details as well. Chapter 4 describes the implementation of the proposed authentication system in Verilog Hardware descriptor language. In chapter 5, simulation waveforms are explained. Results are shown and discussed in chapter6. Chapter 6 concludes the thesis parts 1 with scope of the future work.

Part B contains the internship work done at Fermedicius labs Pvt. Ltd. Chapter 1 provides the introduction to the IoT and various aspects of it, issues in IoT. Chapter 2 provides a brief details about the projects that were done during the course of 6 months of internship at Fermedicius Labs Pvt. Ltd.

Chapter 2

2.1 Literature Review

There are different approaches available in literature that talk about biometric security and authentication. Juels and Wattenberg in [5] talked about the biometric data in general sense and used error correcting code and hamming distance to measure the similarity between two biometrics. Hash function along with some random code word are stored in the database. They discussed about uniformity in distribution of inputs in biometric authentication and other real-world applications. What kind of error correcting codes are suitable for these biometric systems.

The fuzzy vault scheme proposed by Juels and Sudan in [2] considers set difference as the measure of similarity and polynomial interpolation for error correction. This method embeds message into a set to protect it. If anyone else want to extract the message, he should offer another set which is close to the previous one. (LOCK, UNLOCK) was used to keep a document securely in which a fraction information is revealed and will be used during authentication phase to unlock the document. Authentication will succeed only when the biometric input is close to the stored biometric.

Dodis et. al. [22] gave the notion of fuzzy extractor where a strong extraction (such as pair wise independent hash function) is applied often the original biometric is reconstructed to obtain an almost uniform key. Different metrics were elaborated for constructed and analysis Hamming distance, set difference and edit distance.

In Ghaffar et. al. [1], secure face recognition system perform face detection with skin colour detection. PCA (Principal Component Analysis) is used for face recognition and Advanced Encryption Standard (AES) is used for encryption/decryption of data. Generated key is not stored in the database, it is extracted by expanding the submitted user identification.

Davida et. al [10] proposed a system in which error correcting codes were used with cryptographic techniques to achieve the security goal. They proposed a system in which biometric data can be kept in a protected form with some tolerance to the corruption. They computed check bits in the template and stored them along with the hash function.

Sutcu *et al*. [23] used a concept of secure sketch which is generated from a feature vector and a key from robust hashing function. Extracted features from a biometric sample are discretised using quantization. They tried to minimize the entropy loss of the original template by reducing the information stored in the sketch.

Nandakumar *et al*.[3] provided an overview of various biometric threats and protection methods are discussed. Their advantages and disadvantages in terms of security and revocability are also listed.

Yuen *et al*. [24] proposed a face biometric data protection system by using Reed Solomon codes. Biometric data is broken down to integer and fractional parts, integer part is encoded using Reed Solomon algorithm and that coded data along with fractional data are stored in the database. Smartcards are used to store the encoded data along with fractional data.

In [12], Ashish and Sinha used Gabor filtering to discard redundant data from grey image on which morphological operations are performed to obtain minimally connected strokes is performed via ridge stop finding and bifurcation finding. The obtained biometric template is then encrypted to form key.

Biometric recognition system designed in this thesis, takes a more practical approach is used in which speed and security can focused. FPGA implementation takes care of the speed by concurrently performing various modules and SHA-2 which is collision free as of now, is used for securely keeping the information.

2.2 Theory

2.2.1 Error Correction

In any digital communication system (as described in the Figure 2), starting from the process of transmission through to the reception, data is prone to be exposed to errors. These errors can be in the form of noise, interference etc. Poor reception techniques also accounts for the incorrect depiction of the transmitted data. In order to deal with these problems, error detection and correction techniques are used.

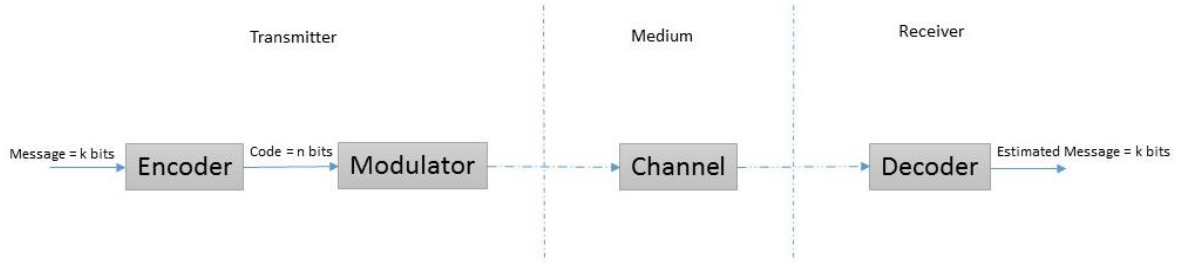

Figure 2 Digital Communication system

Forward error correction technique is used detection as well as correction of the errors introduced in the data. Technique is based on sending some redundant data along with the message that enables the receiver to detect the error and correct it. Usually this is a one –one mapping between message and the code word generated after encoding. Suppose, messages is of k bits and code word is of n bits then $\{0, 1\}^k \rightarrow \{0, 1\}^n$. Obviously there are only 2^k code words will be used in $\{0, 1\}^n$ space.

There are various error detection techniques in literature:

- Repetition system- in this technique, transmitter sends same data multiple times consecutively and at the receiver these data sets are compared. If there is any difference between these data sets then an error is implied. This techniques is not used often as it reduces the data rate of transmission.
- Parity systems- All the data sets are assigned a particular parity (even or odd). At the receiver end all those data sets that do not have the same parity are said to be erroneous. This technique cannot detect even number of errors as in that case that actual parity of the system will remain the same.
- Checksum- A checksum is calculated and sent along with the data. At the receiver, checksum of the received data is calculated and is compared with the transmitted checksum information in the received data, if there is a mismatch, then there is an error. But such a system in not useful if the checksum itself is erroneous.
- Cyclic Redundancy Check- In this system, a generator polynomial is used to calculate the parity data which is added to the original data before transmission. The transmitted code is always divisible by the generator polynomial, if it fails to do so at the receiver end then an error is detected. This is a bit complex technique but works efficiently.

There are few parameters that can used to select or to judge the performance of a particular forward error correction scheme:

- Coding Gain: It is the difference between signal to noise ratio (SNR in dB) of the message bits with and without that particular coding scheme.
- Coding Rate or coding efficiency: It is defined as the ratio of the total length of the message to the length of the code word (k/n). There needs to be a trade-off between code efficiency and error correcting capability.
- Power Utilization (power/bit): It is defined as the amount of power required to send a particular code word.
- Code Complexity: It is used to describe the design time and latency of a coding scheme during encoding and decoding of a message.

There are various Error Correction coding schemes available in the literature, a brief comparison of a few schemes is done by describing them:

- Hamming code: In a hamming encoder, parity bits are inserted into the message bits. These parity bits are calculated in such a way that they produce a fixed parity on different combinations of message and parity bits. At decoder side, those combinations are checked for that fixed parity and accordingly decoder parity bits are set. Binary equivalent of this combination act as an error locator and that error is corrected by flipping the bit. This system can detect up to only 2 error and correct only one error. So this scheme is not suitable for detecting burst errors, which is usually the case.
- Low density parity check code (LDPC): These are the linear block codes, in which message is transformed to the code word by multiplying it with a low density (number of 1's is less) transform matrix. These codes have best coding gain but designing these codes is very complex.
- Turbo code: These codes are merely the convolution codes, so a memory element is required. They are a two stage coding scheme at the decoder, soft decoding followed by hard decoding. Error correcting capability is very good so consequently the coding efficiency is very low. The latency involved is also very significant therefore these codes are not suitable for most of the applications.
- Reed-Solomon code: They are the linear cyclic redundancy non-binary systematic block codes. A generator polynomial is used to generate the parity and the message is shifted by n-k times in order to append the parity to generate the code word. At the decoder, generator divisibility is

used to determine the error. It has a stage wise i.e. error locator stage followed by the error magnitude evaluation stage. These codes have moderate error correction capability but their code efficiency is good and are lesser complex as compared to the above codes. Reed-Solomon codes provide better latency and throughput [16] and are easier to implement in hardware and can correct burst error. Therefore it proves out to be an optimum solution in comparison to the above three error correction codes.

2.2.1.1 Basics of code theory

2.2.1.1.1 Group

Is a set G, with one binary operation (+: G x G –› G) i.e. if the operation is applied on two elements belonging to G then resultant will also belong to G only. And the binary operation is said to be a group if:

- Associativity:

  x+(y+z)=(x+y)+z; ¥ x, y, z € G ….(2.1)

- Identity: There exist one element 0 in G such that,

  x+0=0+x=x; ¥ x € G ….(2.2)

- Inverse: ¥ x € G there exists (-x) € G such that,

  x+(-x)=(-x)+x=0

For example, in {0,1}: Group, + mod 2 ,

1+0=0+1=1

And, (-1) =1 because in modulo 2,

1+1=0 and 1-1=0

2.2.1.1.2 Field

A set F with two binary operations (+, .)is said to be a field if

- Additive property: (F,+) is an abelian group i.e. it follows commutative property.
- Multiplicative property: (F\{0}, .) is an abelian group.
- Distributive property:

  x.y+x.c=x.(y+z) x ¥ x, y, z € F ….(2.3)

A finite field is also called Galois field (GF)

For example F_i or GF(i) having i elements{0,1,2,…,p-1} where i has to be a prime number, with {+, .} operations and mod i is a field:

For $i=2^k$ i.e. GF(2^k) is a finite field if $X^{2k-1} = 1 = X^o$

2.2.1.1.3 Primitive Polynomial

A primitive polynomials is of interest because such functions define the finite fields GF (2k) that in turn are needed to define R-S codes. An irreducible polynomial f(X) of degree k is said to be primitive if the smallest positive integer n for which f(X) divide X^n+1 is n = 2k – 1

For example, $f(X) = 1 + X+X^4$….(2.4)

Here, k = 4 therefore determining whether it

Divides $X^n + 1 = X^{(2m-1)} = X^{15} + 1$, but does not divide $X^n + 1$, for values of n in the range of $1 \leq n < 15$. It can be verified that, $1 + X + X^4$ divides $X^{15} + 1$, and after repeated computations, it can be verified that $1 + X + X^4$ will not divide X^n+1 for any n in the range of $1 \leq n < 15$.

Therefore, $1 + X + X^4$ is a primitive polynomial.

2.2.1.2 Reed Solomon Code

Reed-Solomon or RS codes are linear redundancy non –binary systematic error correcting codes. By (Figure 3) systematic it means that message and parity data are arranged in such a way that the whole message is put together as whole and the parity data is put together as a whole before transmission i.e.

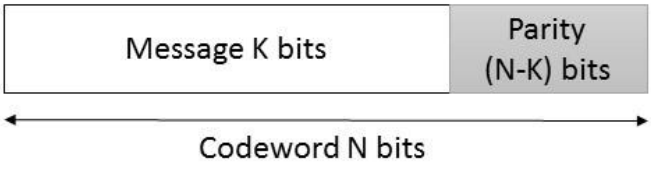

Figure 3 Systematic Representation of Reed Solomon Code

These codes were introduced by Irving S. Reed and Gustave Solomon in 1960s. They have very wide variety of applications ranging from consumer technologies to data transmission technologies to storage systems or even in satellite communication. They are said to be a special case of BCH codes which are binary codes. RS codes are denoted as RS (n,k) , where n is the total length of the code word and k is the message size . They are said to non-binary codes i.e. the value of each symbol of the code word can be chosen to be greater than or equal to 2. An RS (n,k) is said to be valid if :

$$0 < k < n \leq 2^m - 1 \quad \text{....(2.4)}$$

Where m is the number of bits required to represent each symbol.

$$n - k = 2t \quad \text{....(2.5)}$$

Therefore, $t = (n - k)/2$

Where t is the error correcting capability of the code. Out of those 2t, t symbols are used for locating error and t symbols are used to correct those errors.

For Reed-Solomon codes, the minimum distance between two code words so that they can be decoded without any ambiguity is defined by the Hamming distanced (d). It is the number of bits by which any two code words differ and it is,

$$d_{min} = n - k + 1 \quad \text{....(2.6)}$$

Therefore, $d_{min} = 2t + 1$ or $d > 2t$

Reed-Solomon codes is useful in correcting burst errors. It means that in a symbol, if only one bit it containing error or all the bits are erroneous, decoder treats them as a 1 symbol error. In some cases the size of the code word is needed to less than $2^m - 1$, in such cases shortening technique can be used in which padding is done to make that RS(n,k) as a standard full size encoding. After encoding those extra 0s are remove before transmission. At the receiver again padding is done to generate the actual message with padded 0s which later can be removed. BCH codes (binary) provides better error correction capability as compared to Reed-Solomon codes but at the cost of complexity. For instance RS(255,223) has a correction capability of up to 10-15% while, BCH codes have 17% of correction capability. But in case of Reed-Solomon, Galois Field is $GF(2^8)$ which means maximum degree of the polynomial is 7 and in case of BCH Galois field will be $GF(2^8x8)$ which means, higher order polynomial operations will take place costing the complexity of the system.

2.2.1.2.1 RS Encoder

For encoding as well as decoding, generator polynomial G(x) is essential.

$$G(X) = g_0 + g_1X + g_2X^2 + \ldots + g_{2t-1}X^{2t-1} + X^{2t} \quad \text{....(2.7)}$$

The maximum degree of generator polynomial is equal to the number of parity symbols. G(X) has 2t successive power of roots α,

$$G(X) = (X - \alpha^{m0})(X - \alpha^{m0+2})...(X - \alpha^{m0+2t}) \quad \text{....(2.8)}$$

Value of m_0 is either 0 or 1.

In order to encode, message is represented in the polynomial form with the powers of α which represent the data. For instance message is α^2, α^4, $\alpha^6$the,

$$D(X) = \alpha^2+\alpha^4X +\alpha^6X^2 \quad \ldots.(2.9)$$

Systematics representation of block codes require message D and parity P and to create code word C, D has to be shifted to the rightmost position by multiplied it by X^{n-k}. P is generated by dividing the right-shifted D by the polynomial G, remainder of this division is taken as parity.

$$P(X) = X^{n-k} D(X) \bmod G(X) \quad \ldots.(2.10)$$

$$C(X) = P(X) + X^{n-k} D(X) \quad \ldots.(2.11)$$

Above equations imply that C(X) is a multiple of G(X) and representing P(X) in terms of quotient and remainder we get, $X^{n-k} D(X) = Q(X)G(X)-P(X)$

Where Q(X) is the quotient when D(X) is divided by G(X).

From above equation, $Q(X)G(X) = X^{n-k} D(X)+P(X) = C(X)$ ….(2.12)

This implies that C(X) is a multiple of G(X), this statement is very crucial for the proper reception of data.

2.2.1.2.2 RS Decoder

At the decoding end, received data is sometimes corrupted which needs to be corrected. R(X) = C(X) + E(X) is the actual data that is received which has both correct data and error as well. Since here the + operation is modulo 2 which is basically an XOR function, by using the disable mode of XOR operation which is A XOR A =0, we can decode the original data C(X). For this purpose all that is needed is E(X). In order to find E(X), its location as well as its magnitude has to be calculated. Since there can be at most 2t errors, there are at most 2t unknown values, therefore 2t equations are needed to be solved.

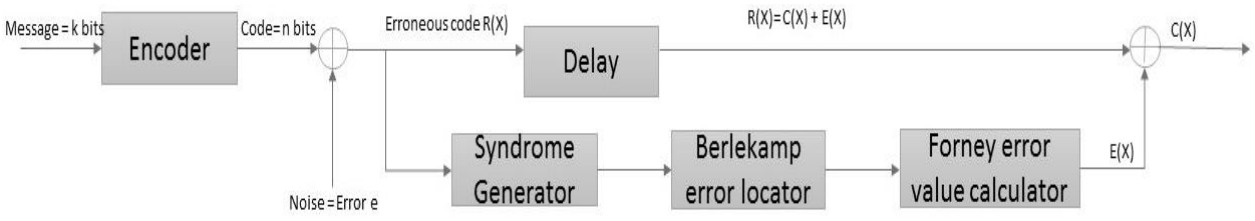

Figure 4 Reed Solomon Decoder

As it is shown in Figure 4, Syndrome generator is used to see whether the received code word has errors or not. Syndrome generator generates a

polynomial of degree 2t and every non zero value of those 2t polynomial will indicate the presence of error.

$$S_i=R(X)|_{x=roots(G(X))}, \text{i.e. } R(\alpha^i) \text{ , } i=1 \text{ to } 2t \quad \ldots.(2.13)$$

$$R(X) = C(X) + E(X) \quad \ldots.(2.14)$$

Since C(X) is a multiple of G(X), $C(\alpha^i)$ is 0

Therefore $S_i=E(\alpha^i)= e_{i1}\beta^i_1 +e_{i2}\beta^i_2 +...+e_{i2t}\beta^i_{2t}$, i=1 to 2t ….(2.15)

$S(X)= S_1x +S_2x^2+\ldots..+S_{2t}x^{2t}$ ….(2.16)

S(X) in above equation is the final syndrome polynomial which will show the presence of error in the received code word, ideally E(X) is 0 therefore, S(X) will also be 0.

Now, to determine the locations of errors in the received cod word, error locator polynomial Λ(X) needs to be generated which is,

$\Lambda(x) = (1+X_1x)(1+X_2x)\ldots(1+X_tx)$ ….(2.17)

Each of these X_i will give the location of the error as each X_i is simply α^i.

In the above equation, degree of the polynomial will be 2t and that polynomial will have some coefficients will be (1, Λ_1, Λ_2….. Λ_t) i.e.

$\Lambda(x) =1+ \Lambda_1x+ \Lambda_2x^2+\ldots..+ \Lambda_tx^t$ ….(2.18)

And $S(x)\ \Lambda(x) = (S_1x +S_2x^2+\ldots..+S_{2t}x^{2t})(1+ \Lambda_1x+ \Lambda_2x^2+\ldots..+ \Lambda_tx^t)$….(2.19)

In the above equation 2.19, s(x) is known but Λ(x) is not known. Solving the above equation by using GZP decoder in which Berlekamp[] algorithm is used, location of the error is determined.

As it can be seen that all coefficients from t+1 to 2t are absent in the equation for Λ(x)

Therefore, the coefficients of $x^{t+1},x^{t+2},\ldots,x^{2t}$ =0. This will give t linear equations and then solving them will give us Λ_i and then roots of Λ(X) has to be determined.

This is done by making a polynomial with Λ_i then finding its roots and inverse of those roots will produce X_i, which will locate the errors in the code word.

Next step is to determine the magnitude of the errors.

$S_1 =Y_1X_1+Y_2X_2+\ldots+Y_tX_t$

$S_2 =Y_1X_1^2+Y_2xX_2^2+\ldots+Y_tX_t^2$

·Therefor $S_i = Y_1X_1^i + Y_2X_2^i + \ldots + Y_tX_t^i, i=1 \text{ to } t$ ….(2.20)

Solving Y_i will give the magnitude of those errors.

Then $e(x) = Y_1X^{i1} + Y_2X^{i2} + \ldots + Y_tX^{it}$, here $X_j = \alpha^{ij}$ ….(2.21)

Adding e(x) to R(x) will give C(X).

Hence the code word C(X) is decoded with errors less than or equal to t.

2.2.2 Linear Feedback Shift Register (LFSR)

An LFSR is a shift register that generates sequences of pseudo random numbers. Seed is used to generate that sequence. Each seed produces a unique pseudo random numbers for a given input sequence. Seed can be of smaller size than the sequence and it needs to be secure as one who has the seed can generate the sequence as an attempt of eavesdropping on the input sequence. The sequence is recurrent i.e. each term is expressed as a function of the preceding terms. Any number of operation can be performed on the sequence terms to make them as random as possible.

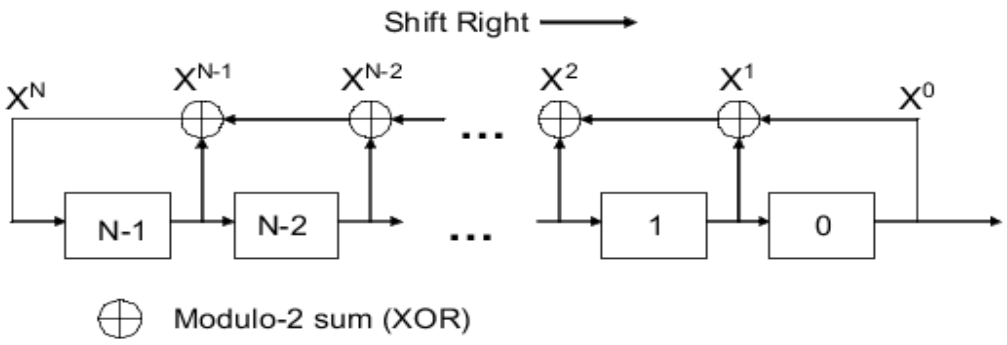

Figure 5 Linear Feedback Shift Register

In the above Figure 5, N bit sequence is generated by successively applying modulo 2 operation on the previous bits.

Initially, Seed (initial sequence) and input are applied to the LFSR. Then there are two ways of producing the sequence, either sequentially like it is shown in the above figure or concurrently by applying parallel operations on the data based on the input values presented. Both of these ways have their own merits, former method save the hardware requirement but takes very long to produce the sequence if the length of the sequence is very long, latter take very less time as all the operations are happening concurrently but the hardware requirement is more for this , since more number of concurrent resources are required. So, based on the application the designing of the LFSR should be done. In This

application Parallel executing LFSR is required because all of the processing takes place after the random sequence is generated. In thesis work, LFSR is used to generate a 223-byte random data which is used by the Reed-Solomon encoder to generate a 255-byte code word which will be used to store input related data in the database.

2.2.3 Secure Hash Algorithm (SHA-2)

Securing the message in any communication system is very vital. Hash functions are the assisting functions in cryptography. Motivation behind developing hash functions was to provide secure signature to the documents of larger size. By using the usual techniques, it was taking very long to generate the key from fixed size block of data. If the data size was huge, then the number of blocks will also be very large which will result in longer time. Hash function overcame this issue by fulfilling following requirements:

- No constraint on the length of the message
- Fixed length output (hash value)

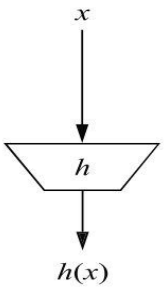

Figure 6 Pre-image Resistance

- Pre-image resistance: Pre-image in this context means the input message and this describes as shown in the Figure 6, the one way ness of the hash function i.e. If H(x) is known then it is impossible to generate x from it.

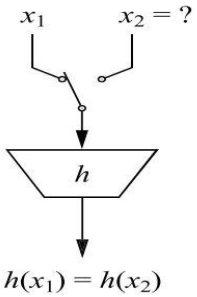

Figure 7 Second Pre-image Resistance

- Second pre-image resistance: This is also called weak collision resistance.

Figure 7 shows that for a give x1 there should not exist any other x2 such that H(x1)=H(x2).

For example, if x1 = Send $10 to A And x2 = Send $1000 to A, if for both x1 and x2,if hash function is same then there will be some cyber felony happening causing incorrect transaction.

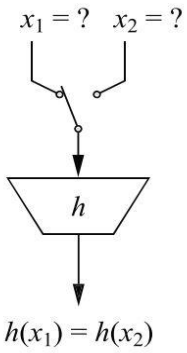

Figure 8 Collision Resistance

- Collision Resistance: Collision resistance is much harder to prevent because x1 and x2, are not specified as shown in the Figure 8. Therefore, if for any two x same hash function is not possible.
  If |x|>>|z|, collision must occur i.e. it will follow "Dirichlet's drawer principle" which states that,
  If there are 10 socks and 9 drawer then collision will occur.

Birthday Paradox:

For weak collision detection (Figure 9), minimum number of people that are needed to be invited such that any one of them will have birthday on a specified given date say June 10 is 365.

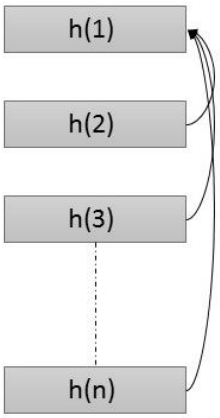

Figure 9 Weak Collision Detection

For collision detection (Figure 10), any two persons having birthday on same date is evaluated.

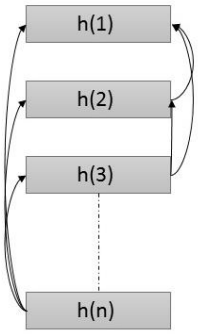

Figure 10 Collision Detection

P (no collision will occur among 2 persons) =1-1/365

P (no collision will occur among 3 persons) = (1-1/365) (1-2/365)

P (no collision will occur among t persons) = (1-1/365) (1-2/365)… (1-t/365)

After taking t=23, it is observed that probability of collision is around .507 i.e. 50%. And for t=40, probability of collision is 90% so in this case the total number of person require is very less as compared to weak collision resistance. In general form, total number of bits required to detect a collision is,

$t \approx 2^{(n+1)/2}(\ln(1/(1-\lambda)))^{1/2}$….(2.22)

Where λ is probability for at least one collision.

Using the above results, it comes out that for n=256 i.e. 256 bit hash value function, only around 129 bits are required.

SHA256: Secure hash function 256 operates in the same MD5, SHA-1 does.

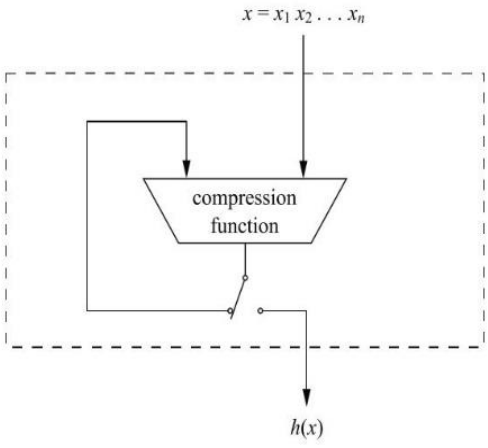

Figure 11 Block Diagram of Merkel-Damgard Construction

It uses Merkel-Damgard construction, which is an iterative process and message is provided as the input and output (hash function) is obtained at the last iteration of the compression.

Figure 11 depicts the basic diagram of the construction.

The basic operations involved in the hash function generation are:

- Message is padded to make it a multiple of 512-bits.
- The group the message into 512-bit long blocks M_1, M2,...,$M_{n.}$

Message blocks are processed one at a time,

$H^i = H^{(i-1)} + C_{M(i)}(H^{(i-1)})$….(2.23)

Where C is the compression function and + is mod 2^{32} addition (normal addition but ignoring the carry generated on the MSB), $H^{(0)}$ is the initial value of the hash function which is 32 bit word. High level diagram of the SHA256 is shown in Figure 12.

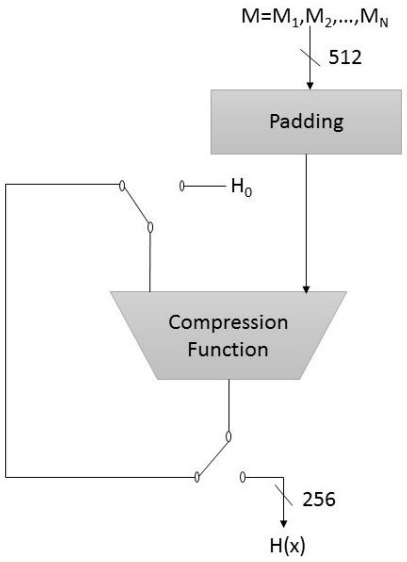

Figure 12 Block Diagram of SHA2 (256)

Preconditioning of the message takes two steps:

- Padding: If a message is x and its length is l bits then

  k= 512-64-1-l

  = 448-(l+1) mod512

  K is the number of 0s that will be appended to the message.

  After appending k, binary equivalent of number of bits l is appended at the end in 64 bits. 1 will be inserted after the message and before appending k bits.

So, in total message will be a multiple of 512.
Ex: If message is “abc” then l is 8*3=24
k=448-(24+1) mod512
k = 423bits (number of 0s)
Now the 512 bit message will be
01100001 01100010 01100011 1 000...00(423 bits) 00…011000(64 bits).

- Grouping the message into N 512 bit blocks. Those 512 bits divided into 32 bit sub-blocks. This is done because each round treats on 32bit words for operation.

The main block has a total of 64 rounds and operations in each takes following routes:

Repeat for i=1 to N

- Initialize registers a,b,c,d,e,f,g,h with the (i-1)st intermediate hash function value for(initial value when i=1)
  Here a,b,…h are 32 bit registers.
  $a = H_1^{(i-1)}$
  $a = H_2^{(i-1)}$
  .
  .
  .
  $h = H_8^{(i-1)}$

- Now applying compression function of SHA256
  Repeat for j=0 to 63
  Compute ch(e,f,g),Maj(a,b,c),$\Sigma_0(a)$, $\Sigma_1(e)$, and W_j

$ch(a,b,c)=(a\&b) \wedge (\sim a\&b)$ ….(2.24)

$Maj(a,b,c)=(a\&b)\wedge(a\&c)\wedge(b\&c)$ ….(2.25)

$\Sigma_0(a)= S^2(a)\wedge S^{13}(a) \wedge S^{22}(a)$ ….(2.26)

$\Sigma_1(a)= S^6(a) \wedge S^{11}(a) \wedge S^{25}(a)$ ….(2.27)

$\sigma_0(a)= S^7(a) \wedge S^{18}(a) \wedge R^3(a)$ ….(2.28)

$\sigma_1(a) = S^{17}(a) \wedge S^{19}(a) \wedge R^{10}(a)$ ….(2.29)

R^n is shift right by n bits and S^n right Rotation by n bits

$T_1 = h + \Sigma_1(e) + ch(e,f,g) + K_j + W_j$ ….(2.30)
$T_2 = \Sigma_0(a) + Maj(a,b,c)$ ….(2.31)
$h = g$ ….(2.32)
$g = f$ ….(2.33)
$f = e$ ….(2.34)
$e = d + T_1$ ….(2.35)
$d = c$ ….(2.36)
$c = b$ ….(2.37)
$b = a$ ….(2.38)
$a = T_1 + T_2$ ….(2.39)

Figure 13 gives a systematic view at the operations happening in compression function.

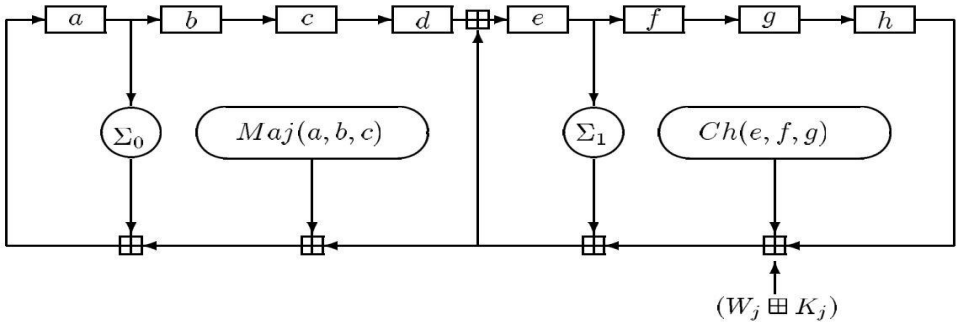

Figure 13 Detailed view of Compression Function

⊞ Denotes mod 2^{32}

- Calculating intermediate hash function $H^{(i)}$

  $H_1^{(i)} = a + H_1^{(i-1)}$
  $H_2^{(i)} = b + H_2^{(i-1)}$
  .
  $H_8^{(i)} = h + H_8^{(i-1)}$ ….(2.40)
  $H^{(N)} = (H_1^{(N)}, H_2^{(N)}, \ldots H_8^{(N)})$ is the hash function for message M.

Message blocks W_i(i=0 to 63) are computed using a message scheduler as shown in the Figure 14

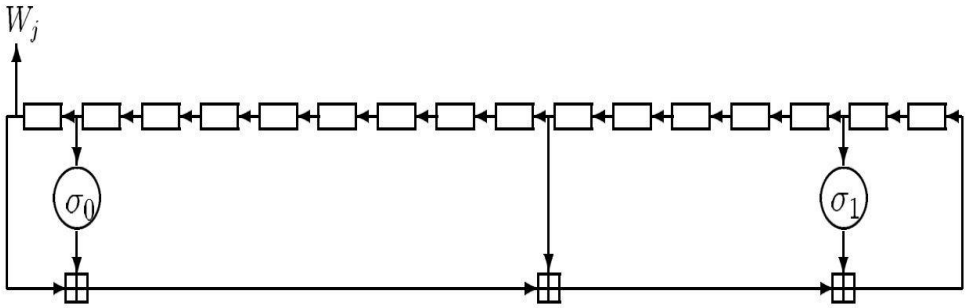

Figure 14 Generation of Message Blocks

$W_i=M_i^{(j)}$ for i=0 to 15 and for i=16 to 63,

$$W_i= \sigma_1(W_{i-2})+ W_{i-7} + W_{i-16} + \sigma_0(W_{i-15}) \quad \ldots.(2.41)$$

$W_i$is used as a key for the individual rounds and their value is managed by the scheduler to produce 64 32 bit W_i words from the available 16 32 bit message words.

Chapter 3

Proposed work

In this section, proposed work inspired from fuzz commitment by (Ari Juels, 2013) *et al*., in which they perform the fuzzy commitment of the biometric (finger print) is presented. A face recognition system on FPGA that does the recognition in a fuzzy way is designed. Fuzziness means the biometrics of the same person that do not match with the enrolled biometric when presented in the authentication phase. Human face presented for authentication is subject to change in numerous ways from the stored biometric like beard, spectacles, posture, angle at which the image is captured, luminance, hair style ,expressions etc. there are various algorithms which focus on nullifying all these changes and provide a reliable face matching system. But there has not been much focus on the security of the biometrics. Biometric information once revealed, can grant access of the secured data to the attacker and he can misuse it.

In this thesis work, various algorithms are combined to provide a reliable and secure face recognition system. There are basically two phases:

1. **Enrolment Phase**: In this phase, the input biometric (face) m 200x255 is presented and a random number r will be generated using LFSR (linear feedback shift register) which will be fed to the Reed Solomon Encoder to generated code word z. After encoding, Hash function of that number will be generated using SHA-2(256) algorithm. And the difference u between m and z will be stored in the stored in the memory along with hash function of z.
2. **Authentication Phase**: In this phase, biometric m` is presented at the input for authentication of the corresponding person. The input m` is treated as the corrupted sample and the difference u is use to find the varying code word z`. Now the Reed Solomon Decoder is applied on this code word z` to correct the errors in the input biometric and generate r. If the errors in the input image are beyond a particular threshold then it would not be possible for the reed Solomon decoder to generate the correct r from z`. Once r is obtained, the hash function of r is generated and then the hash function matching is performed. If the match occurs, then the person is authenticated as the valid person. Otherwise, that person is denied the access and is asked to try again.

In this whole system, biometric m is not saved in the memory. Instead the hash function of a random number is stored which is used to interpret the input data and store the difference value u. Any attacker will not be able to attack and get the information about the user whose biometric is saved in the memory since, everything depends on the randomly generated number which is securely saved in a hash function. This hash function will reveal the value of r only when m` is close enough to m. Closeness is measured in terms of the capability of the reed Solomon code to correct the errors which depends on the code word length and the message length.

If the difference between code word length and message length is increased, this capability of correction increases but the coding efficiency decrease and vice- versa. So, there is a trade-off between coding efficiency and capability of error correction. Random number used there is of length 223 byte which is fed to the Reed Solomon Encoder to generate a 255 byte number. More the length of random number generator the better it is, and 223 byte number proved to be a fairly good enough number. SHA-2 provides a secure 256 hash function for the given 255 byte input data. The whole system depicted in the Figure 15 is implemented on the FPGA development board Zybo (Zynq board).

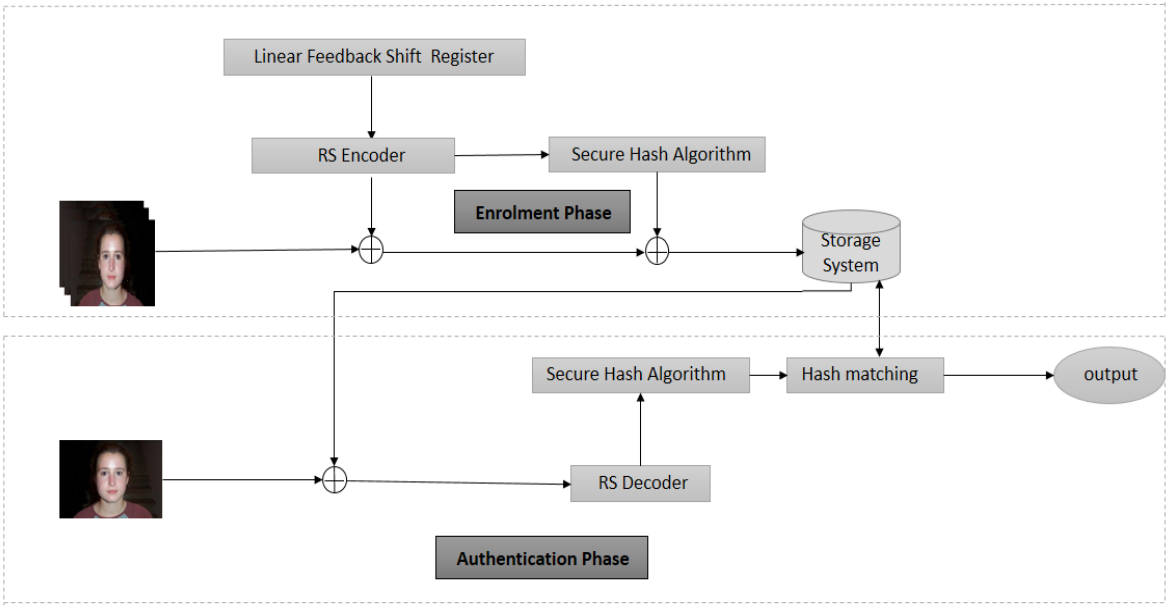

Figure 15 Proposed System for Recognition

3.1 Enrolment

At the time of Enrolment,

- Linear Feedback Shift Register (LFSR) is initiated, which feeds the 223-byte random number to Reed Solomon Encoder.

obj= RSencoder(n,k); ….(3.1)

C=gen(obj,msg); ….(3.2)

Herein equation (3.1), n=255-byte and k=223-byte and obj is 32-byte long. In equation (3.2), C is the final code word comprising both message that is random number and parity data.

- Hash function algorithm takes the 255-byte (2040-bits) as input and generates a 256 bit key or hash function which is unique for each input code word.

$H : \{0,1\}^n \rightarrow \{0,1\}^l$ ….(3.3)

Here in equation (3.3), n (code word length) is 2040-bit long and l (hash function output length) is 256-bit long. h(z) is a one way function producing a unique 256-bit number for each code word.

- If the input data during authentication is close to the saved data then only after some manipulation the hash function will produce the same key. Otherwise, data mismatch will occur and user will not be authorized.
- F(z,m)=(h(z),m-z) ….(3.4)

After the generation of hash function, F(z,m) is computed and saved in the database for each image presented during enrolment phase. Equation (3.4), h(z) is stored along with ‘m-z’ which subtracts z from each row of m, size of the resultant is 200x255 as well.

Note: Data directly related to image m is not stored in the database at any point rather, everything depends on the random number r which is produced at the time of each enrolment. So the risk of exposing the user information is minimized. This crucial process of using r for almost every manipulation, provides the security of biometric.

3.2 Authentication

At the time of Authentication,

- The input data is used to unlock the random number stored securely inside a hash function.
- At input, m` is presented which manipulated as shown in the equation (3.5) to generate z.

$z_e = f(m` - u)$ ….(3.5)

Where u= m-z, substituting the value of u in (3.5) we get

$z_e = f(m` - (m-z))$

$= f((m`-m)+z)$

Let m`-m= Δt, representing the error between the input sample and the stored image.

$z_e = f(\Delta t + z)$, taking $\Delta t+z=z`$ we get,

$z_e = f(z`)$ ….(3.6)

Here function f(z`) is performing Reed Solomon decoding to generate $z_e$ from z`.

- This recreation corrects at most 16byte error in the code word z` because for Reed Solomon,

$2t=n-k;$ ….(3.7)

Where t is the error correcting capability of the code and n and k are code word length and message length respectively.

Since n=255-byte and k = 223-byte, from (vii) we get 2t= 32

Which gives us t =16-byte.

- After generating z_e which is 255-bytes long, its hash function h(z_e) is generated is matched with the stored hash function in the data base.
- If h(z_e)=h(z), the person with m` is authorized as a valid user. Otherwise, a mismatch will occur and user will again have to input the biometric.

Note: There needs to be an absolute symmetry between the hash function algorithms used during enrolment and authentication phase otherwise, the results will be incorrect. And the parameters set for the Reed Solomon encoder and decoder should also be symmetric for proper error correction.

3.3 Reliability and Resilience of the Proposed work

A face recognition system is said to be reliable if it follows two norms:

1. Probability of guessing the message or face should be less than 0.5.
2. It should be impossible for the attacker to unlock z and have a match by entering incorrect biometric data.

The first norm focuses on the possibility of guessing the value of r to generate z if he is able to generate z then he can find x which corresponds to that z since u=m-z is publically available. In order to fulfil this norms, random number generator. Random number used here is of 223-bytes i.e. 1784-bits which implies there are 2^{1784} possible numbers and therefore it is nearly impossible to the value of r. The second norm focuses on the onewayness of the hash function i.e. for any x there should not exist any x` which will result in h(x) = h(x`). This property of hash function is referred to as strong collision protection and secure hash algorithm (SHA-2) ensure this property.

Resilience of the system depends on the ability of a system to tolerate the errors in the incoming corrupted data (m`). Reed Solomon provides correction of up to 16bytes in each code word in our system as we are using RSencoder(255,223) scheme.16bytes of error can be a distributed or a bulk error, system will treat indifferently. This error is evaluated by calculating the hamming distance (D) the code words which tells the number of bits that are different in those two words. If e is the error introduced at the input then a resilient system should satisfy the following equation:

$$h(f(m+e-u)) = h(f(m+e-(m-z))) = h(f(z+e)) = h(f(z)) \quad \ldots.(3.8)$$

There is a trade-off between code efficiency (k/n) and ability to correct errors (2t=n-k). If k is increased then code efficiency will increase but the room for parity bits reduces therefore error correction capability reduces. In this work, RS (255,223) code (message length is 223-byte and code word is 255-byte) is used as it provides 16-byte error correcting capability in each row of the image (200x255) input during authentication.

Chapter 4

Hardware Implementation and Simulation

Hardware level block diagram is shown in the Figure 16. Some conditioning is required at the image sensor to make images compatible with the following modules. Figure 16 has 3 major parts and their implementation is explained in detail in the following text.

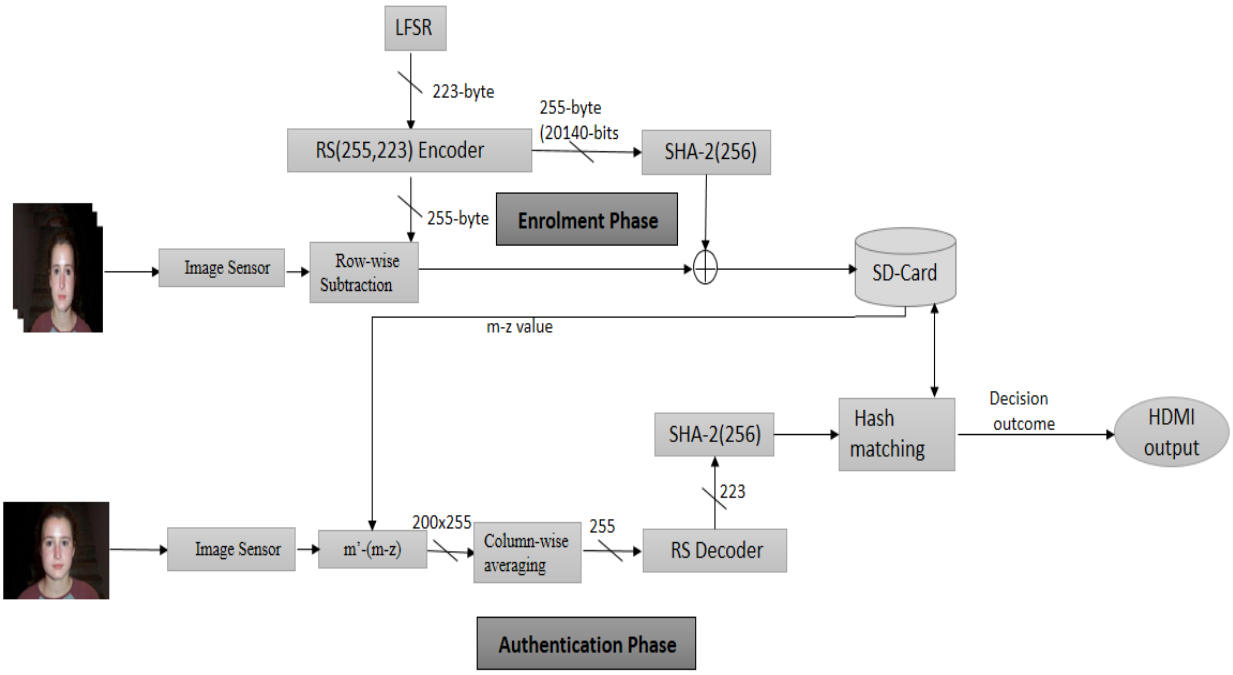

Figure 16 Block Diagram of Implemented Design

4.1 SHA2-256

Figure 17 shows the core block of SHA-2 in which a block of 512-bit is fed and an output digest of 256-bit is obtained which is the hash value corresponding to that block. At each positive edge of the clock clk, Initialis value is checked if it is high then message block is fed to the internal registers for internal manipulation described in 2.2.3. 512-bit message is divided into 16 32-bit blocks and remaining blocks are generated by applying operations of the available blocks as shown in Figure 14.

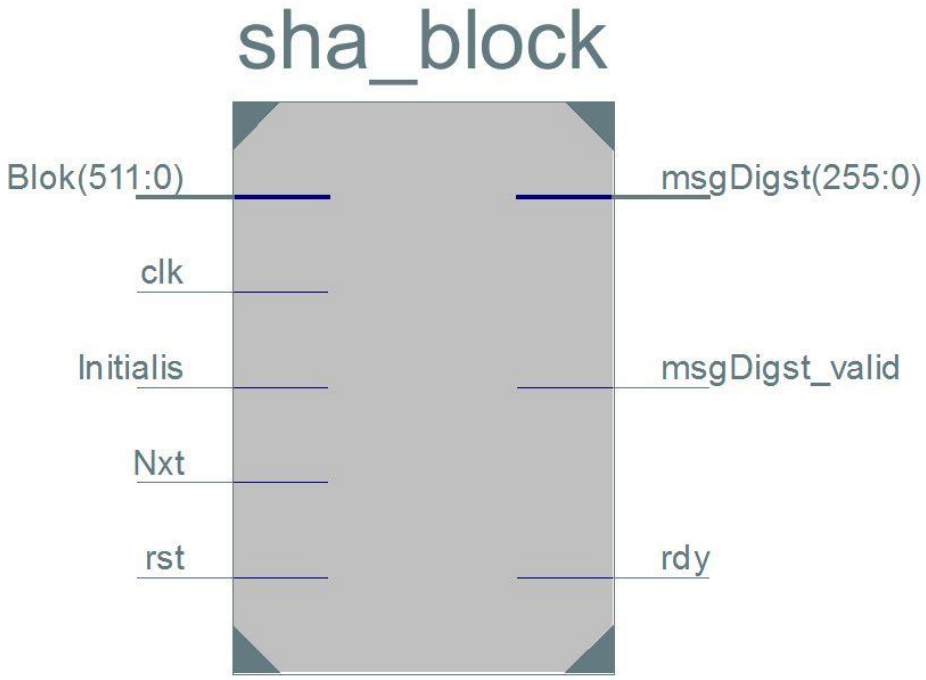

Figure 17 Core block of SHA-2

Signals that are used in the top module of SHA-2 are described below:

- Initialis: When high, data block is loaded in to internal registers to generate word.
- Nxt: This signal loads the next 32-bit message block for word generation.
- rdy: Tells that the module is ready is receive the 512-bit message for word generation.
- msgDigst_valid: Is high when 256-bit hash value is generated.
- rst: Is an active low signal. When low, block works normally. When high, all the registers are reset.

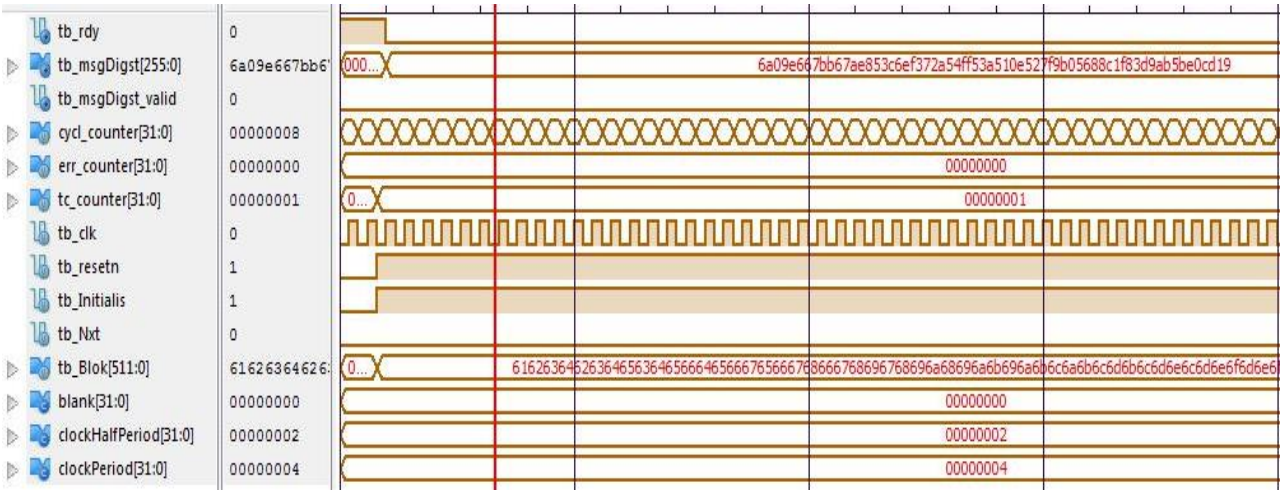

Figure 18 SHA-2 Waveform

Figure 18 shows the simulation results of SHA-2 in which, a 256-bit digest is generated for corresponding message data. As we can see in the figure, msgDigest_valid becomes 1 whenever digest is ready. At the rising edge of the clock word generator will generate a 32 bit word and that word is fed to the compressor for further operations. Therefore after 64 internal cycles, final message digest will be generated which be a collection of 8 32 bit registers forming a 256bit hash value.

Figure 19 shows the RTL Schematics of SHA-2.As it can be seen that there are two internal modules sha-wrd_mem generator block and sha_ki_const generator block. sha_wrd_mem generator converts 512-bit message block into 64 32-bit blocks using the operations explained in Figure 14.

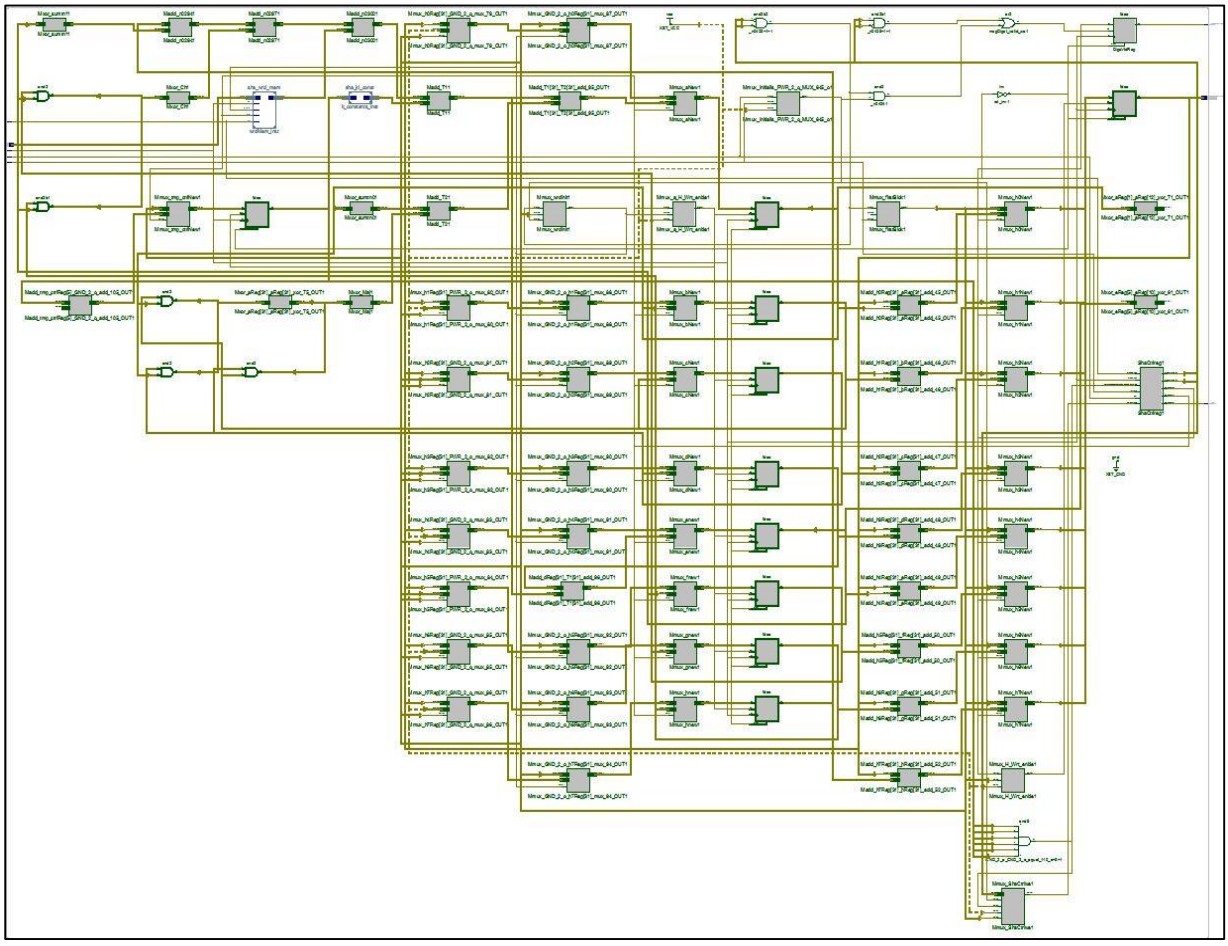

Figure 19 RTL Schematic of SHA-2

Figure 20 shows the simulation results of sha_wrd_mem generator. It can be seen that at the rising edge of the clock, conversion from 512-bit message block to 64 32-bit w(word) block takes place. First 16 blocks are same as the message

data, 17-64 blocks are generated internally by performing operations on the given data.

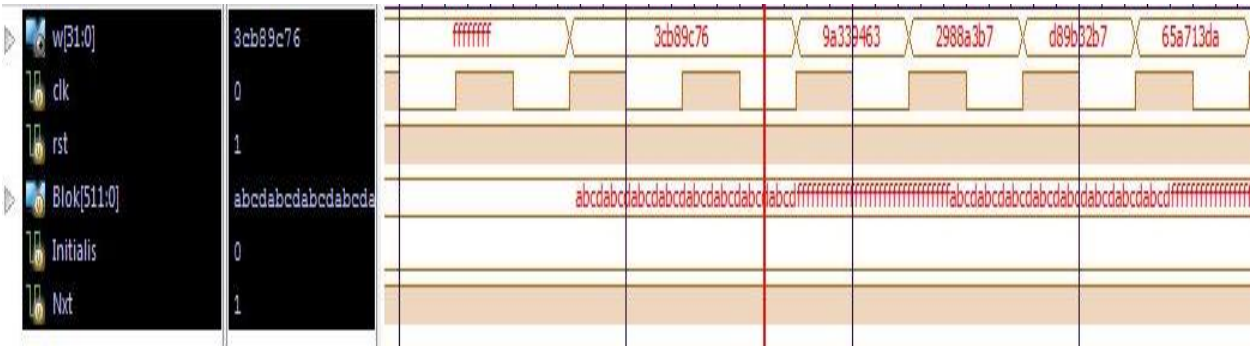

Figure 20 Message to Word conversion

Sha_ki_const module (Figure 21) generates a 32-bit constant value for a given address.These values are provided by NIST(National Institute of Standards and Technology) and are used to generate hash values as shown in Figure 13. Module sha_ki_const has 64 such values, each correspond to an address When the module is called address is incremented from 0 to 63 for each round in hash function generation

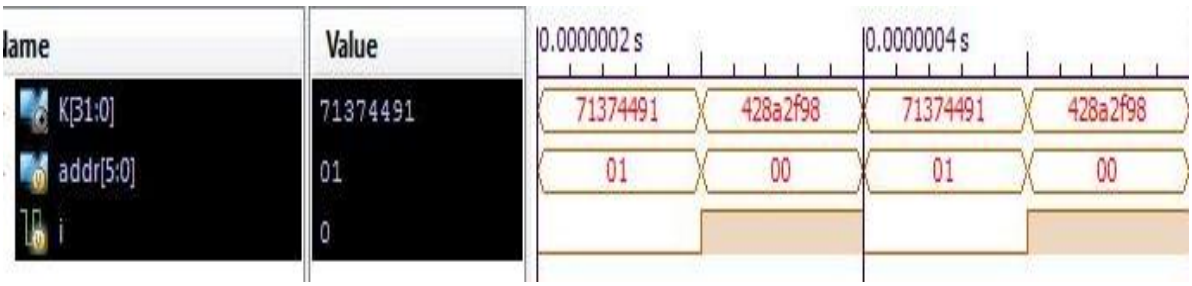

Figure 21 Constant Ki generation

4.2 LFSR (Linear Feedback Shift Register)

Figure 22 shows the block level diagram of LFSR. It has two internal signals, initial value and tap input. Initial value is loaded into the output while reset (rst) is low. When rst becomes high, at each clock rising edge, output y is calculated. Initial value, tap input and output y, all are 1784-bit long for the purpose of feeding y to reed Solomon encoder and decoder to generate a code word. Initial value and tape input are set to a default value which could be made generic or external input ports to make them customization at the user end. Making them generic would make the system more secure as user can change them time to time. Here the randomness of LFSR is 2^{1784} which means, probability of

guessing a code word is $1/2^{1784}$ which is nearly impossible in the current scenario.

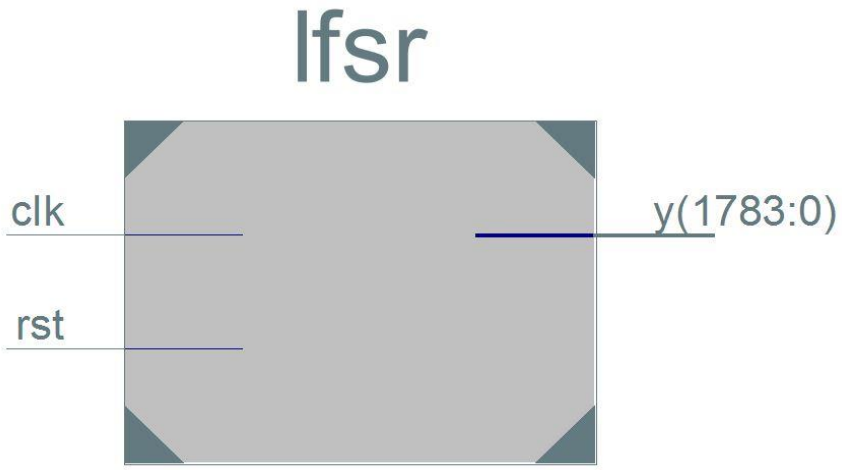

Figure 22 LFSR Top module

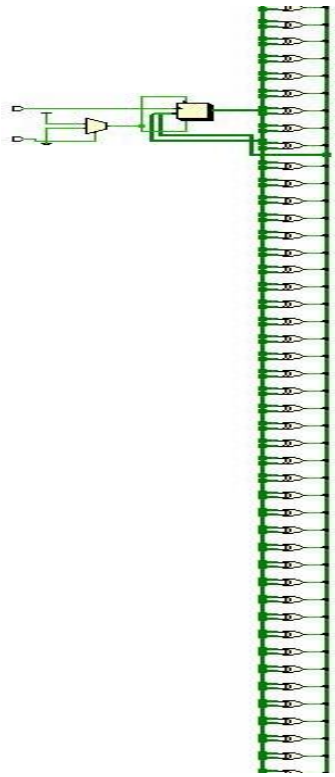

Figure 23 RTL Schematic of LFSR

Figure 23 shows the RTL level schematic of LFSR, as it can been seen that a lot of flip-flops (hardware) are used, this is done in order to generate the random output as soon possible because other modules in the complete system depend on the output of the random number generated by the LFSR.

Figure 24 shows the simulation results of LFSR. As it can be seen, at each positive edge of clock clk, output y (1784-bit) is generated which will be called in the Reed-Solomon module for code word generation.

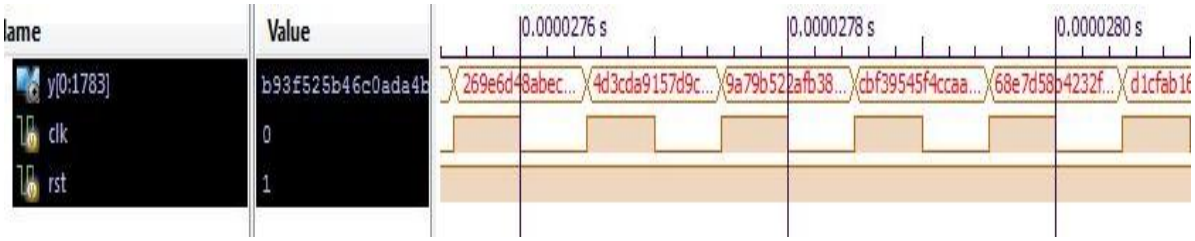

Figure 24 Simulation results of LFSR

4.3 Reed-Solomon Code

In order to implement Reed-Solomon Encoder/Decoder, Galois field operations need to be performed. Galois field addition is simply modulo2 addition or a logical XOR operation. Galois multiplication is not that simple. After multiplication, field of the product should not change. In the designed system, $GF(2^8)$ is used. $X^8+X^4+X^3+X^2+1$ is chosen as primitive polynomial. And the generator polynomial is

$g(x)= (x-\alpha)(x-\alpha^2)(x-\alpha^3)\ldots.(x-\alpha^{32})$

Where α is the root of the generator polynomial.

Encoder

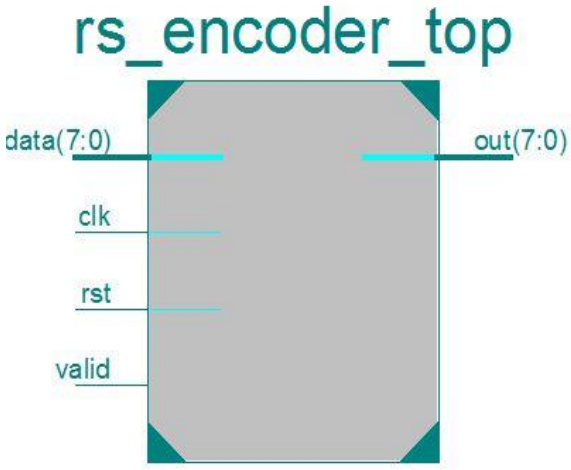

Figure 25 RS Encoder Block

The code word will be represented as

$c(x)= x^{(255-223=32)}d(x) + p(x)$

Where $p(x) =x^{(255-223=32)}d(x)/g(x)$

Table 1 lists the elements of field GF(2^8). It has 256 elements on GF(2^8), out of which 255 elements are non-zero. Since the primitive polynomial is $X^8+X^4+X^3+X+1$, elements with power greater than 7 can be derived using primitive polynomial expression.

X8=X4+X3+X+1 will be used to reduce the polynomial multiplication to the highest degree of 7.

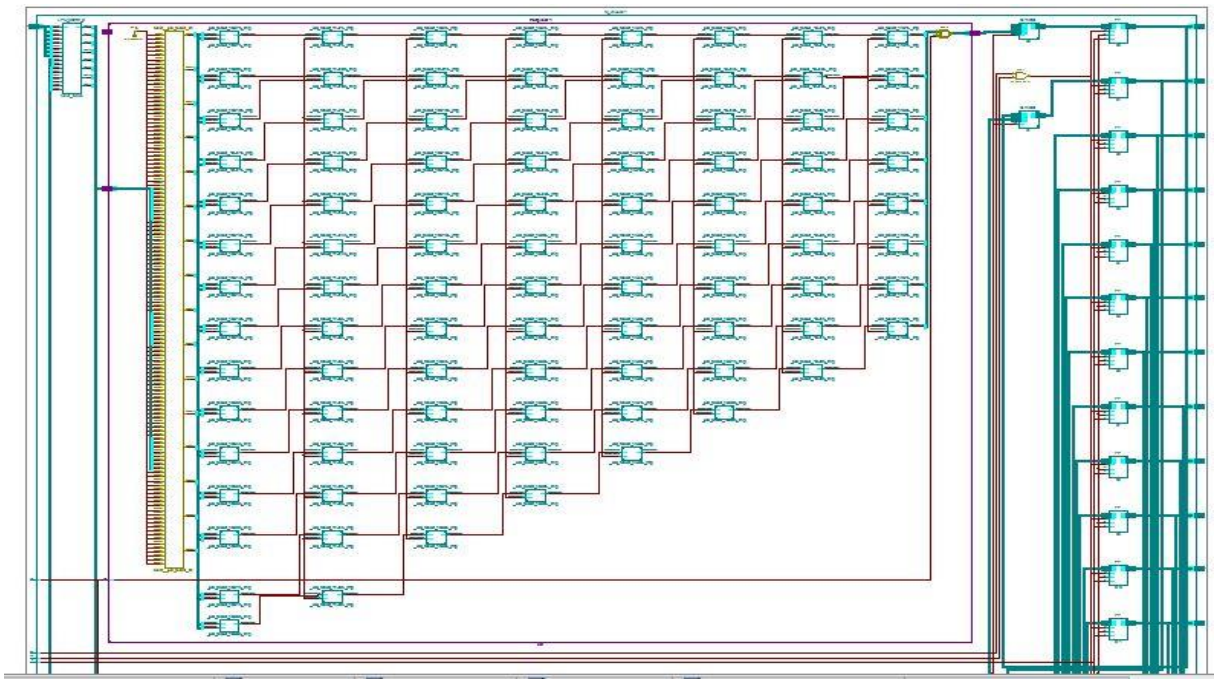

Figure 26 RTL Schematic of GF Multiplier

As discussed in 2.2.1.2.1, Galois multiplier is a complex operation. Therefore it acquires major part of the RTL design as visible from Figure 26.

Table 1: Elements of field GF(2^8)

Power $(\alpha)^i$	Polynomial Form	Binary Form	Decimal Form
0	0	00000000	0
α^0	1	00000001	1
α^1	α	00000010	2
α^2	α^2	00000100	4
α^3	α^3	00001000	8
α^4	α^4	00010000	16
α^5	α^5	00100000	32
α^6	α^6	01000000	64
α^7	α^7	10000000	128
α^8	$\alpha^4 + \alpha^3 + \alpha^2 + 1$	00011101	29
α^9	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha$	00111010	58
α^{10}	$\alpha^6 + \alpha^5 + \alpha^4 + \alpha^2$	01110100	116
α^{11}	$\alpha^7 + \alpha^6 + \alpha^5 + \alpha^3$	11101000	232
α^{12}	$\alpha^7 + \alpha^6 + \alpha^3 + \alpha^2 + 1$	11001101	205
α^{13}	$\alpha^7 + \alpha^2 + \alpha + 1$	10000111	135
α^{14}	$\alpha^4 + \alpha + 1$	00010011	19
…	…	…	…
α^{253}	$\alpha^6 + \alpha^2 + \alpha^1 + 1$	01000111	71
α^{254}	$\alpha^7 + \alpha^3 + \alpha^2 + \alpha^1$	10001110	142

$g(x)= (x-\alpha)(x-\alpha^2)(x-\alpha^3)….(x-\alpha^{32})$ will give

$g(x)=45+ 216x+ 239x^2+ 24x^3+ 253x^4+ 104x^5+ 27x^6+ 40x^7+ 107x^8+ 50x^9+ 163x^{10}+ 210x^{11}+ 227x^{12}+ 134x^{13}+ 224x^{14}+ 158x^{15}+ 119x^{16}+ 13x^{17}+158x^{18}+ x^{19}+ 223x^{20}+ 164x^{21}+ 82x^{22}+ 43x^{23}+ 15x^{24}+232x^{25}+ 246x^{26}+ 142x^{27}+ 50x^{28}+ 189x^{29}+ 29x^{30}+ 232x^{31}+ x^{32}$

Where each element represents the value of coefficient from g_0= 45 to g_{32}=1

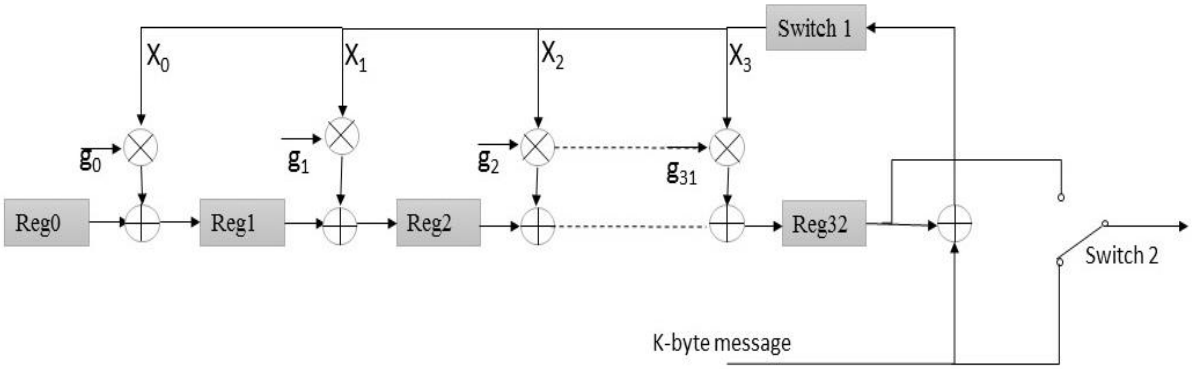

Figure 27 LFSR operation of Reed-Solomon Encoder

Figure 27 shows the LFSR operation, which performs the encoding part of Reed-Solomon.

- Switch 1 is closed during the first k (223) clock cycles in which the tap sequence (input message) is shifted in to the LFSR. Switch 2 is connected to the input which is forwarded to the output directly for first 223 clock cycles.
- Once the 223th message symbol passes the switch, switch 1 is opened and Switch 2 is then connected to the output of the LFSR to get parity output.
- In remaining (32) clock cycles, parity data is generated one by one. Generator polynomial g(x) coefficient values are Galois multiplied with the message symbols loaded earlier. Output of multiplication are then added with the register values.
- Once reg32 generates a value, it is forwarded to the output through switch 2 and the same value is shifted by 32 bits and is sent for the Galois multiplication and addition.
- The result will be a code word of the form $c(x) = x^{32}d(x) + x^{32}d(x) \bmod (g(x))$. RTL schematic of the RS Encoder is shown in Figure 28.

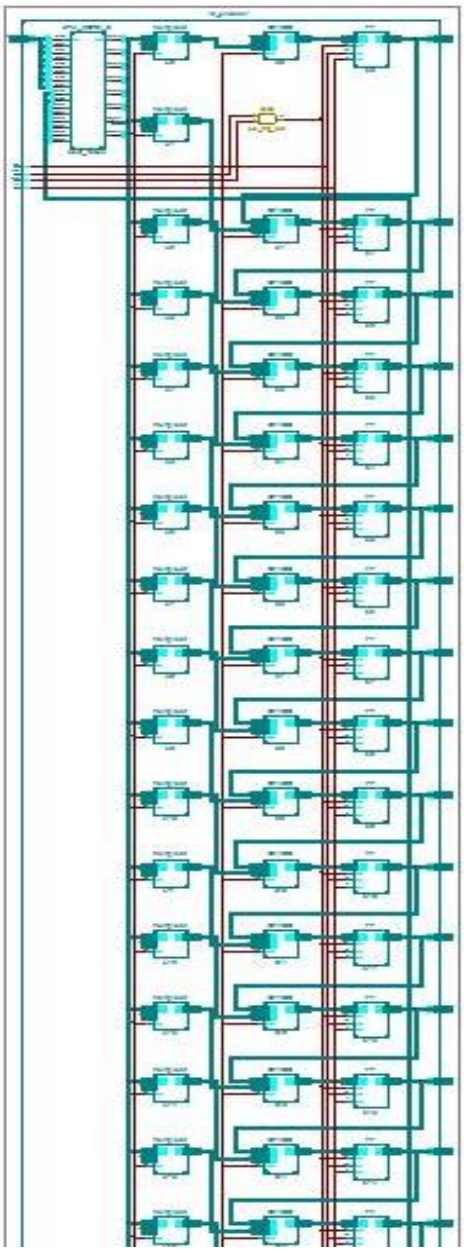

Figure 28 LFSR structure of the RS-Encoder

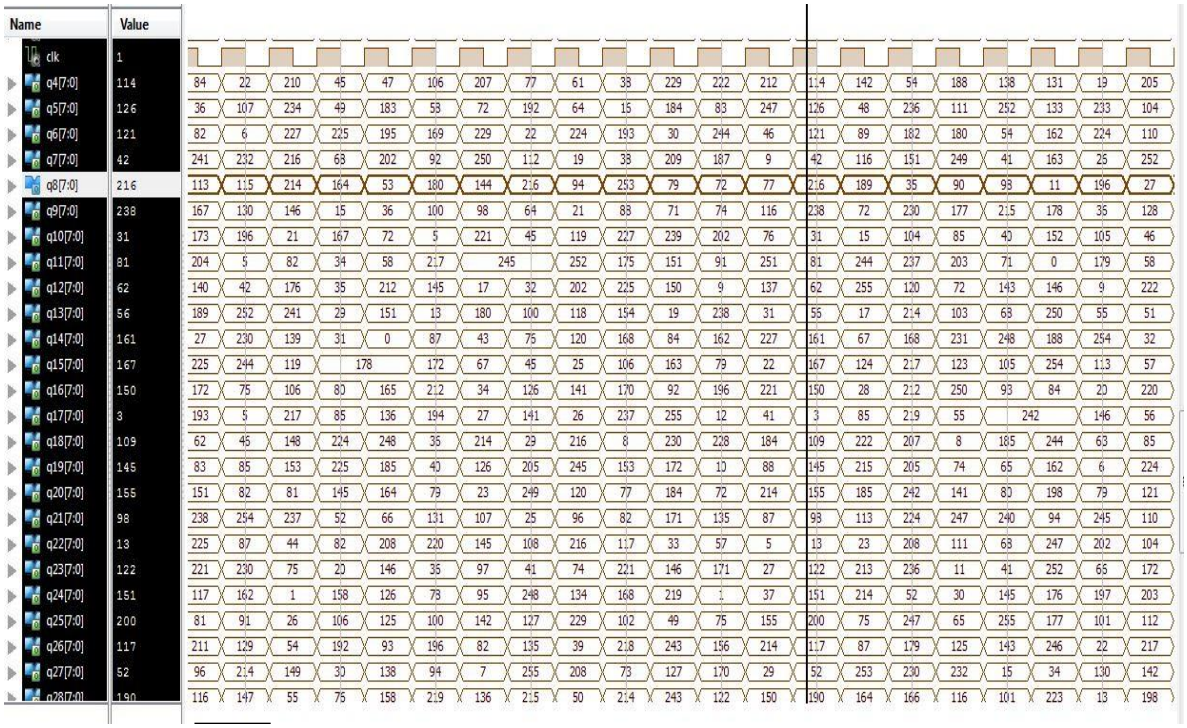

Figure 29 Simulation results of RS- Encoder

Figure 29 shows the RS-Encoder output q0-q31 getting generated at positive edge of clock. Initially, 223 clock cycles are for data and after that quotient of Galois multiplication after getting added to previous register value is produced. Next 32 clock cycles are for q0-q32 which represents the parity data of the code word.

Chapter 5

Results

A training set of 400 images from CALTECH open source face database is used. Each image is conditioned to 200x255 resolution and there are 20 individuals with each having 20 image containing different angle, illumination and expression. MATLAB simulation of the whole recognition process was run and FAR (False Acceptance Ration) was found to be 0% and false rejection ration depends on the Reed Solomon's error correction capability (t) as discussed in chapter 2.Table 2 shows the results in comparison with present work or the earlier work. If the error or variation is within the limit then FRR is 0%. In real time applications, FRR can be maintained either by increasing the training set or by compromising with the security of the database because in that case in order to improve t, k has to be reduced an C (code word) will also reduce in size causing lesser randomness, even if we do this at some point further reduction in k will not improve FRR as the image quality will start to suffer. So, there needs to a trade-off between security and resilience. Following provide the synthesis results obtained for the design from Vivado 16.2 for FPGA and Design Compiler for ASIC design each module.

Table 2. Comparison with other Systems

Metric	[11]	[18]	[10]	[21]	[19]	Proposed
FAR (%)	.1	0	<.01	.85	1.03	0
FRR (%)	.8-1.6	3.5	<.1	3.52	4.1	0 (subjective to t)

Table 3-5 shows the hardware utilisation of FPGA board. As it can be scene, SHA-2 is taking the major part of the hardware available on the board. There are only 2-bit XORs performed in LFSR on a 223-byte data. Therefore the amount of registers and the number of XORs used by LFSR is large.

- SHA-2: FPGA Results:
  FPGA Device - xc7z010clg400-2 (ZYBO)

Frequency =154.667 MHz
Area (Gate Count) =4407;
XOR gate: 3 input 32-bit =5; 2 input 32-bit =1

Table 3: Resource Utilisation of FPGA by SHA-2

Component	Used	Available	Utilisation (%)
Slice Registers	1832	35200	5.2
LUTs	1942	17600	11.03

- LFSR:FPGA Results:
  FPGA Device - xc7z010clg400-2 (ZYBO)
  Frequency =866.664 MHz
  Area (Gate Count) =4586
  XOR gate: 2 input 2-bit=1002

Table 4: Resource Utilisation of FPGA by LFSR

Component	Used	Available	Utilisation (%)
Slice Registers	1790	35200	5.09
LUTs	509	17600	2.89

- Reed-Solomon Encoder: FPGA Results:
  FPGA Device - xc7z010clg400-2 (ZYBO)
  Frequency =164.842 MHz
  Area (Gate Count) =4069
  XOR gate: 8 input 16-bit=29; 2 input 32-bit=203; 2 input 8-bit=62

Table 5: Resource Utilisation of FPGA by RS Encoder

Component	Used	Available	Utilisation (%)
Slice Registers	256	35200	.73
LUTs	285	17600	1.62

- ASIC Results of SHA-2, LFSR and RS Encoder:

  Technology library used: 32nm

  Tool: Synopsis Design Vision

Table 6: Performance Analysis of ASIC

Metric	SHA-2		LFSR	RS Encoder
Power(µW)	Total	646.8857	433.3536	533.6733
	Dynamic	204.4403	19.3269	90.2292
	Leakage	442.4449	414.0366	443.4442
Area(um^2)	22134.989066		22494.587138	18496.579459
Cell Count	4147		1784	4233

Table 6 shows the performance analysis of different function blocks of the system for Application Specific Integrated Circuit (ASIC). Power, Area and Cell Count are measured and as it can be scene that LFSR is taking huge amount of area as it has a large number register in use. SHA-2 dissipates maximum Power (µW) as it has much more dynamic power as compared to other blocks and RS-Encoder has maximum cell count.

Chapter 6

Conclusion

Security and Reliability of a face recognition system were addressed and a design was implemented that will improve both the factors. An end to end Face Recognition System was designed and implemented in Verilog HDL. In the presented work, database was restricted by including all the types of variations possible in the face template. Only the frontal face were used for enrolment as well as authentication. According to the requirement, LFSR should produce output at very high processing speed. In the design, the maximum frequency was found to be 866MHz. This module has more resource utilisation but enhances the overall speed of the system. Onewayness of Secure Hash Function-2 provides the security from collision .i.e. no attacker can impersonate any enrolled user. Reed-Solomon Code improved the reliability aspect of the system by providing error correcting capability. In the designed system, 16-byte per code word could be corrected. The overall system provides a secure and reliable for face recognition system.

Future Work

Design of an algorithm that will tolerate error patterns related to face biometric Optimization can be applied to the individual blocks to increase speed. Better recognition techniques along with the given system could be employed to use a better database which will include all sort of variations in the facial templates.

Bibliography

1. A. Abdel-Ghaffar, M. E. A. H. A. K. M. M. A. A.-A., 2008. A Secure Face Recognition System. *IEEE*, pp. 95-99.
2. A.Juels, M., 2002. A fuzzy vault scheme. *IEEE Int. Symp. on Information Theory.*
3. Anil K. Jain, K. N. A. N., 2008. Biometric Template Security. *Journal on Advances in Signal Processing.*
4. Anon., n.d. *http://www.fingertec.com/images/w_brochure/facerecognition_e.html.* [Online].
5. Aqib. Al Azad, M. H. I. R. R., 2011. Efficient Hardware Implementation of Reed Solomon Encoder and Decoder in FPGA using Verilog. *International Conference on Advancements in Electronics and Power Engineering*, pp. 117-121.
6. Ari Juels, M. W., 2013. *A Fuzzy Commitment Scheme*, Bedford,New York: Springer.
7. Christof Paar, J. P., 2010. *Understanding Cryptography.* New York: Springer.
8. Franssens, A., 2008. *Analysis of Reed Solomon error correcting codes,* Enschede, The Netherlands: s.n.
9. Freire, M. R., 2008. *Biometric Template Protection in Dynamic Signature Verification,* Nebrija: UNIVERSIDAD AUT´ONOMA DE MADRID.
10. George I. Davida, Y. F. B. J. M. R. P., 1998. On the Relation of Error Correction and Cryptography to an Offline Biometric based Authentication System. *IEEE Symposium on Privacy and Security.*
11. Karthik Nandakumar, A. K. J., 2008. Multibiometric Template Security Using Fuzzy Vault. *BTAS(Biometric Theory, Application and Systems).*
12. M.M Ashish, G. S., 2016. Biometric Template Protection. *International Science Community Association.*

13. Malenko, M., 2014. *Implementation of Reed-Solomon RS(255,239).* Macedonia, International Conference on Applied Innovations in IT, pp. 43-47.

14. Mangtani, M., 2016. Implement Reed Solomon Encoder/Decoder on Spartan FPGA. *Journal of Research in Engineering and Applied Sciences*, pp. 199-204.

15. NIST, 2015. *nistpubs/FIPS/NIST.FIPS.180-4.pdf.* [Online] [Accessed 3 2017].

16. Norbert Wehn, S. S. P. S. T. L.-E. M. A., 17 October 2014. *Advanced Hardware Design for Error Correcting Codes.* s.l.:Springer.

17. Shu Lin, D. J. C., 1983. *Error Control Coding: Fundamentals and Application.* Eaglewood Cliffs, New Jersey: Prentice-Hall.

18. T.A.M.Kevenaar, G. M. d. A., 2006. FaceRecognition with Renewableand Privacy Preserving Binary Templates. *Philips Research.*

19. Telgad R L, D. P. D. S., 2014. Combinational approach to score level fusion for Multimodal Biometric System by using Face and Fingerprint. *IEEE transactions on Recent Advances and Innovations in Engineering,* Volume 9, pp. 224-240.

20. Thangaraj, E.-C. T. b. D. A., 2012. *http://nptel.iitm.ac.in.* [Online] [Accessed May 2017].

21. Xiaoxue wang, Y. z. H. y., 2015. bimodal biometric system based on face and fingerprint using decision level fusion. *IEEE transcations on electronics information and emergency,* Volume 7, pp. 21-36.

22. Y. Dodis, L. R. A. S., 2004. Fuzzy extractors: How to generate strong key from biometric and other noisy data. *Springer-Verlag*, pp. 523-540.

23. Yagiz Sutcu, Q. L. ,. N. M., 2007. How to Protect Biometric Templates. *IEEE Transaction on Information and Security*, pp. 503-511.

24. Yuen, Y. C. F. a. P. C., 2006. Protecting Face Biometric Data on Smartcards with Reed-Solomon. *IEEE Computer Society.*

PART – B

Internship work at Fermedicius Pvt. Ltd.

Chapter 1

Introduction

A network of smart devices in which devices communicate with each other and perform tasks with minimal or no human interference of humans in order to provide assistance to the humans and try to make their life easier is termed as Internet of Things (IoT). As it is estimated that by 2020, fifty billion gadgets will be associated with Internet, that means, each person on earth will have more than 5 smart devices on or with them. A device in the Internet of Things can be smart phones or PCs or any man-made devices that can have a unique identity (e.g. IP Address, RFID, etc.) and they must have the capacity to exchange information among themselves. Figure 30 shows the IoT paradigm as a result of convergence of different visions.

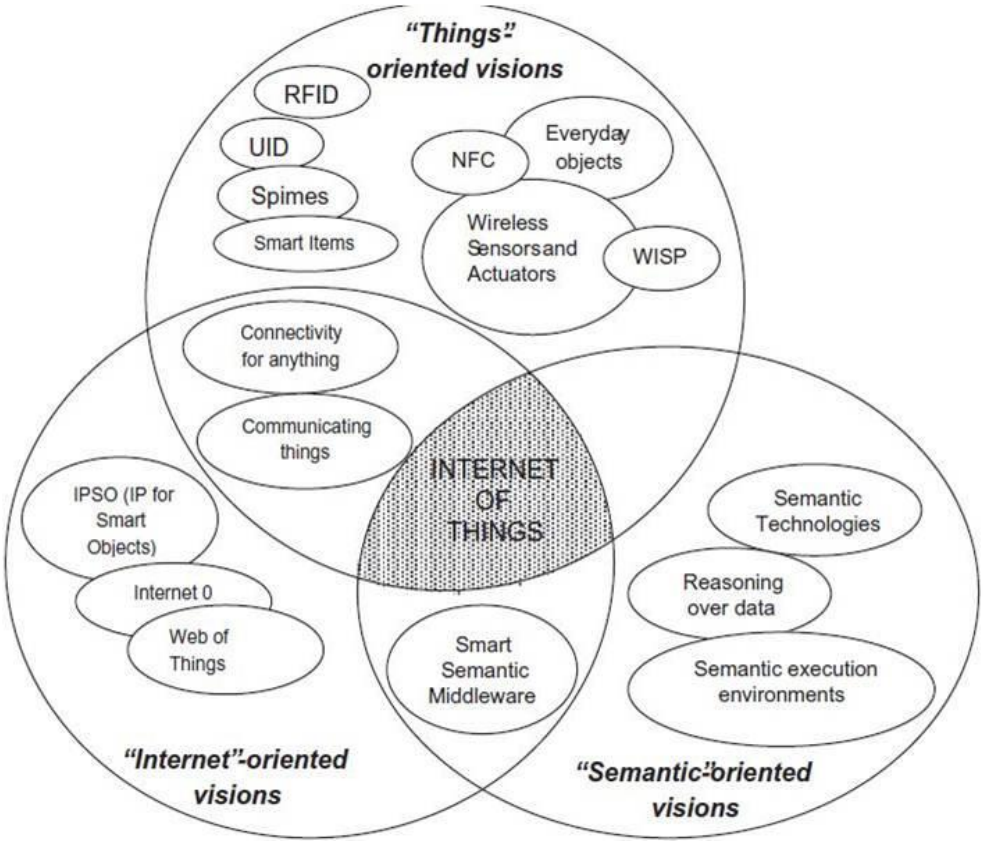

Figure 30 Internet of Things Paradigm

There are three aspects of IoT:

- Things oriented vision: Comprising of all the things associated in an Internet of Things for ex. RFID, Sensors, Actuators, NFCs, Smart items etc.
- Internet oriented vision: This aspect provides the connectivity between devices or things. Ex. IPSO (IP for Smart Objects), web of Things etc.
- Semantics oriented vision: This aspect provides the middleware between things and internet. All decision making, smartness is taken care by this vision.

IoT is going to take very important part in creating smart cities, smart homes and intelligent network between things. Especially in India, IoT is going to play key role in Digital India Campaign. The biggest challenge of the IoT is that it is going to cover different hardware devices to communicate with each other e.g.

communication between a washing machine and a smartphone, communication between a PC and a door's lock and this way we can have infinite combinations and we don't have an effective technology yet which can provide a common platform for such huge number of devices to communicate with each other.

Issues in IoT are:

- Standards: Communication and protocol standardization needs to be done to provide a compatible network for all.
- Mobility: Scalability and Adaptability to heterogeneous technologies is not yet provided.
- Transport Protocol: Obsolete congestion control mechanism to deal with huge amount of traffic built by the data flowing between devices.
- Authentication infrastructure and proper resource distribution and utilisation is required.

Figure 31 shows a general architecture of IoT system comprising things, intra-network, inter-network, back-end services to manage data and do analytics.

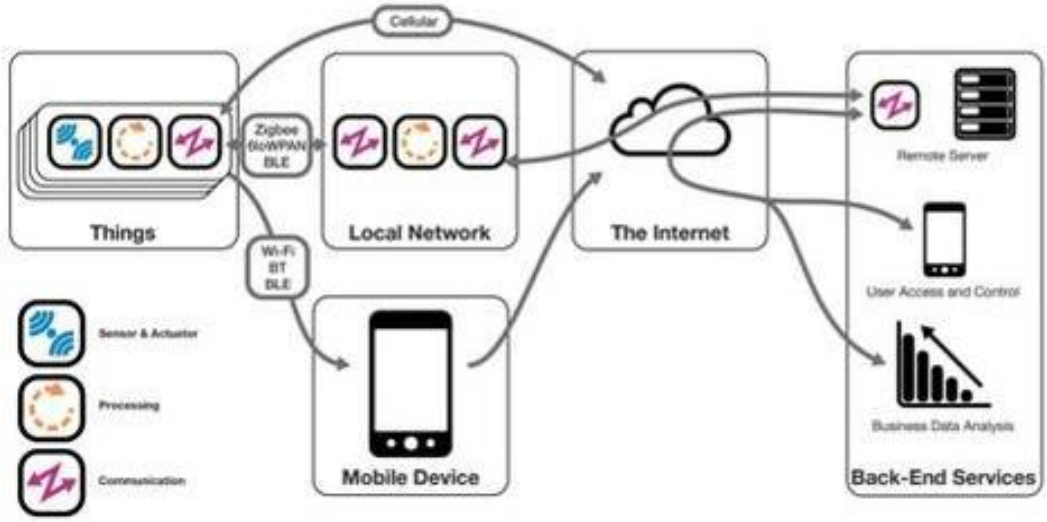

Figure 31 IoT Architecture

Chapter 2

Project Profile at Fermedicius Pvt. Ltd.

2.1 Taco

Detachable IoT system in food packaging to accomplish last mile traceability. System can track the food product in real time, send and store the data in cloud server and map the data to a unique ID generated for each customer and product. Specifications of the said system are:

- Any information that relates to the origin, time of processing, freshness etc. should be available when ever user try to access it.
- Near field communication, QR code type of method should be used in order to provide feasibility to the system because all these methods are available in smart phones and an application could be developed to interact with the system.

Figure 32 illustrates the concept of TACO in which a real time link is established between database and user through open platform.

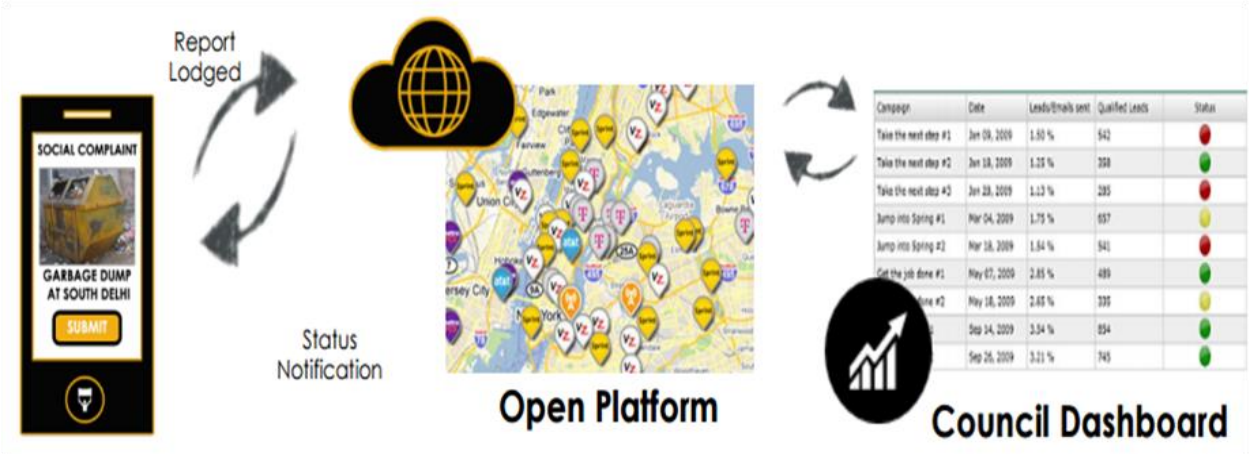

Figure 32 Illustration of TACO

2.2 Smart Base

Design and Development of detachable device that can interact with a computing device and provide product information, notifications and real-time tracking. Device has a RFID scanner to read RFID tag and identify a product and send the information to the said computing device. It has indicators like LEDs, sound, vibrators for notification and alarm. Also has a display unit and load cells for real-time tracking and display purpose. The project will provide a

modular solution to track, identify and notify users about the product. It has following functionalities:

- Bluetooth/ Wi-Fi enables the device to communicate with the mobile phone application or act as standalone device to interact directly with the database.
- TFT screen to communicate and take inputs from the user.
- Notify the users when to eat and what to eat.
- Notify the user when food products expire.
- RFID information helps in keeping track of the origin of the product and provides interesting and knowledgeable information to the users.

2.3 Smart Processing Facility

To design a novel processing scheme consisting of a Centralized Processing Facility (hereafter referred to as Hub and where primary processing happens) and Distribution Centres (hereafter referred to as Spokes and where secondary processing and sale happens). The said processing scheme has to achieve two results:

- Build products in a modular way to meet personalization needs of consumers (primary processing at hubs, secondary processing and packaging at spokes)
- Achieve unit Economics by optimizing procurement, resource utilization, demand forecast, waste reduction and inventory management

The table 1 summarizes the components in the processing scheme (hub and spoke) with the functionality and specifications of IoT system

Table 1 Components of Hub and Spoke adapted in the smart processing system

Part	Functionality to be achieved by IoT	Specifications
HUB	(I) Forecast for procurement of raw produce (II) Sensory networks to monitor the time decay of raw produce (III) RFID Network to track the movement of produce from one unit to another	(a) Raw material Inventory Management: Use sensors to monitor the inventory and send the data real-time to central system (b) To accumulate the demand from spokes and based on the information from Inventory Management System, to generate indent for procurement (c) RFID Network: Design a network of RFID tags and Reader / Writer for a

Part	Functionality to be achieved by IoT	Specifications
		processing plant to track the consignments going in / out of a processing unit (d) For each processing unit (Washing, Pressing, Packaging etc.), displaying Current Load vs Max Load
SPOKE	(I) Forecast demand products (II) Sensory network to monitor the inventory of products	(a) Product Inventory Management: Continuously monitor or count the available products in the spoke's inventory. (b) System to place order if the inventory reach below threshold limit

Figure 33 shows Process and Instrumentation diagram of the Central Processing Unit where the process of distribution to Spokes is done.

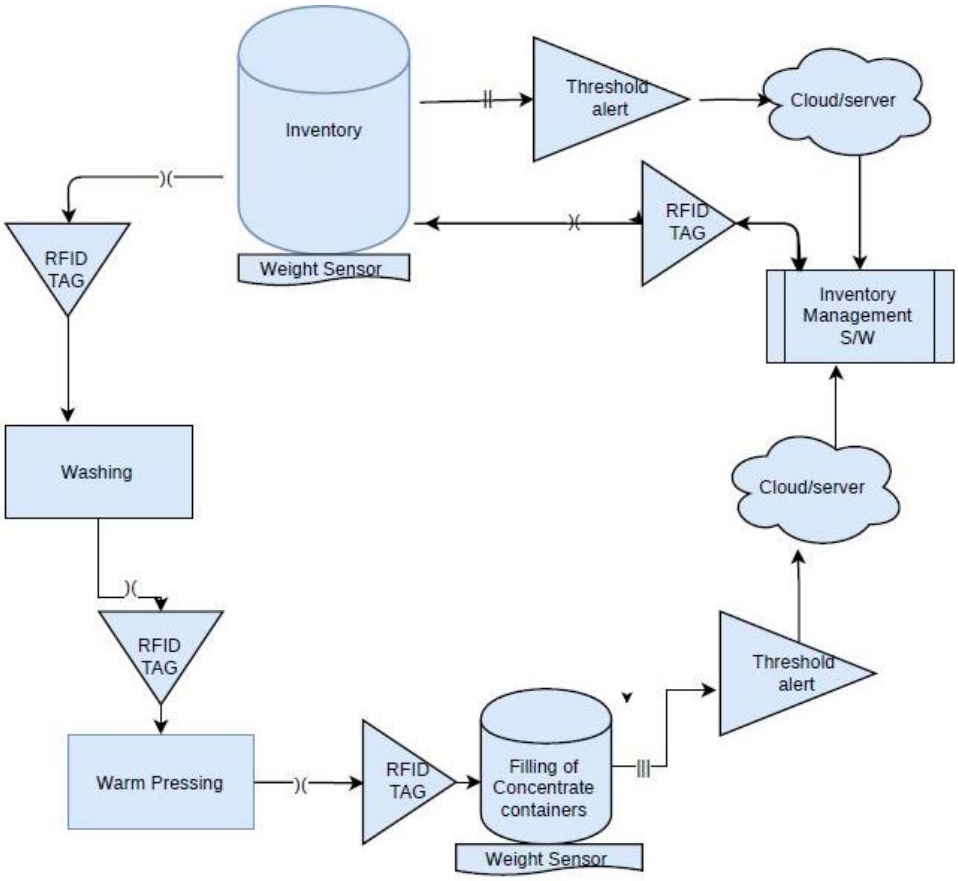

Figure 33 P&ID for Central Facility