International Journal of Electrical and Computer Engineering (IJECE)
Vol.2, No.2, April 2012, pp. 214~222
ISSN: 2088-8708

Iris Feature Extraction and Classification using FPGA

Babasaheb G. Patil, Nikhil Niwas Mane, Shaila Subbaraman
Department of Electronics Engineering, Walchand College of Engineering, Sangli, India
e-mail:babasaheb.patil@walchandsangli.ac.in, mane.nikhil@rediffmail.com, shailasubbaraman@yahoo.co.in

Article Info

Article history:

Received Dec 11th, 2011
Revised Mar 1st, 2012
Accepted Mar 15th, 2012

Keyword:

Finite state machine.
Jacobi transformation
Singular value decomposition
FPGA

ABSTRACT

An approach of singular value (SVD) of a (mxn) 2-D matrix has been popularly used by researchers for representing a 2-D image by a set of less than or equal to n values sequenced in descending order of which a subset of only first few values which are significant is treated as a set of features for that image. These features are further used for image recognition and classification. Though many papers as reviewed from literature have discussed about this implantation using software/MATLAB approach, rarely a paper appears on hardware implementation of SVD algorithm for image processing applications. This paper presents the details of a hardware architecture developed by us to implement SVD algorithm and then presents the results of implementation of this architecture in the Xilinx field programmable gate array Virtex5 to extract the features of an iris image. A comparison between the feature values extracted by MATLAB and those obtained by hardware simulation using Xilinx ISE tool indicates a very good match validating the hardware architecture. A hamming distance classifier using appropriate threshold values stored in ROM is used to classify the iris images.

Corresponding Author:

Babasaheb G.Patil,
Departement of Electronics Engineering,
Walchand College of Engineering ,
Vishrambag, Sangli 416415,
Maharashtra (India)

e-mail :patilbg@rediffmail.com, babasaheb.patil@walchandsangli.ac.in

1. INTRODUCTION

Intelligent personal identification based on biometrics measurement has become very popular in recent years for security purpose. Facial features, thermal emission, iris, gait, voiceprint, gesture, palm-prints, fingerprints, hand-written signature, hand geometry etc. are some of the means for personal identification. Iris recognition is one of such biometric method which is advantageous from variability, stability, uniqueness and security point of view for the reasons given below. Iris is the only internal human body organ that is visible from the outside, thus well protected from external modifiers. The possibility of finding an iris equal to another one is considered to be null. Even the two irises of the same individual do not match. The iris pattern does not change throughout the user's whole life. It is impossible to modify surgically without any risk for the vision. The physical response to light provides a suitable way to test the aliveness of the user (avoiding the use of synthetic eyes). Yet, the human iris is relatively simple to image and is done in a non-intrusive way.

Number of iris recognition algorithms is proposed in the literature such as Independent Component Analysis (ICA), Singular Value Decomposition (SVD), Characterizing key local variation etc. [1] to extract iris features and to propose competitive learning mechanism to recognize iris pattern. Singular value decomposition is simple and provides highly stable results even under changing lightning conditions. Due to large degree of parallelism associated with it, it is a proper candidate for hardware and hence for FPGA

implementation. The analytical details of SVD along-with the details of the architecture designed and developed are presented in this paper.

2. IRIS RECOGNITION

Figure.1 shows the four main steps in iris recognition system, which consists of Image Acquisition, Image Segmentation, Feature Extraction and Classification.

2.1 Image Acquisition

The very first step of Iris Recognition is image acquisition. High resolution camera under certain lightning conditions is used for grabbing the images. Still or video black and white images are generally used to avoid problems regarding dilatation of the pupil. It is necessary to obtain several captures of the iris to assure aliveness and quality of the capture.

2.2 Image Segmentation

Main objective of segmentation is to remove redundant information, namely the pupil segment and the part outside the iris (sclera, eyelids, skin). This is done by cropping the unnecessary information after detecting the Pupillary Boundary and the outer iris edge.

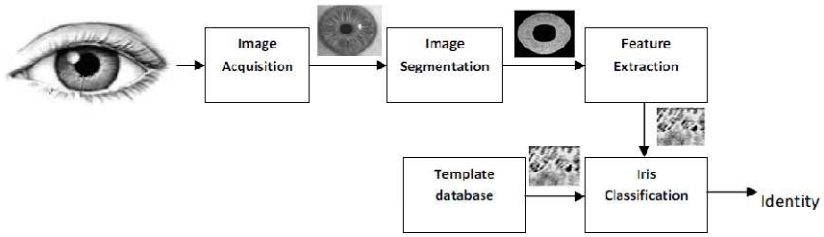

Figure 1. Iris Recognition System

2.3 Feature Extraction

Feature extraction step is used to create a biometric template. The biometric template provides a normalized, efficient and highly discriminating representation of the feature, which can then be objectively compared with other templates in order to determine the identity. This step forms the heart of this paper and is explained in detail in the next section.

2.4 Iris Pattern Matching

Iris matching step is used to match known templates with unknown templates using mathematical distances. If the distance between two patterns is less than some threshold value, then it indicates a good matching between the corresponding patterns. This paper concentrates only on the last two stages viz. feature extraction and iris pattern matching stages.

3. FEATURE EXTRACTION BY SINGULAR VALUE DECOMPOSITION

A process of feature extraction is followed after the process of image segmentation. Due to large redundancy in the segmented image, the cost associated with extracting valuable features from segmented image is quite exorbitant in terms of heavy computational burden, computational time and memory storage. The singular value decomposition algorithm reduces time complexity of iris recognition from $O(n^3)$ to $O(n)$ by using Jacobi transformation. We have used this method for Eigen value computation. Jacobi method offers large degree of parallelism in computation besides providing accurate results even with fixed point arithmetic. This fact leads to fairly simple implementation of SVD algorithm in FPGA.

3.1 Singular Value Decomposition

Singular Value Decomposition is a powerful tool for decomposing the iris basis matrix. SVD exposes the hidden geometry of the matrix. In this paper SVD is used as a dimensionality reduction tool. The basic operation of SVD relies on the factorization of an mxn matrix ($m \geq n$) into three other matrices in the following form:

$$A_{m \times n} = U_{m \times m} \cdot S_{m \times n} \cdot V^{T}_{n \times n} \qquad (1)$$

where the superscript '*T*" denotes the transpose. *U* is an *mxm* orthogonal left unitary matrix, *V* is an *nxn* orthogonal right unitary matrix and S is an *mxn* diagonal matrix with $S_{ij}= 0$ if i ≠ j and Sii ≠ 0. For a given matrix *A*, matrix *S* is unique. The two important aspects to be noted here are:

1. Matrix element of S is zero everywhere except in the main diagonal. This leads to reduction in the dimension of the input pattern from a matrix *mxn* to only a vector of *n* elements.
2. Only the first *k* elements out of *n* elements, when arranged in descending order, contain substantial information, and the vector tail without significant loss of information can be cropped out, leading to further reduction in the dimension of the vector representing 2-D *mxn* matrix.

By re-arranging (1) we get,

$$S_{mxn} = U^{T}_{mxm} \cdot A_{mxn} \cdot V_{nxn} \qquad (2)$$

In the above equation , if *A* is a square matrix, then the left and right unitary matrices, *U* and *V* are derived from Jacobian rotations on matrix *A* iteratively. Considering this aspect, the above equation can be modified to an iterative equation as,

$$A^{i+1} = J^{i^{T}} \cdot A^{i} \cdot J^{i} \qquad (3)$$

Here, *J* is the Jacobi rotational matrix. As the iteration number *(i)* tends to infinity, off diagonal elements of the iterated matrix *A* tend to zero. The singular values of the original matrix *A* are then obtained by taking the square root of diagonal elements of the transformed matrix.

3.2 Construction of Jacobi Matrix

Consider a 2x2 matrix A as below

$$A = \begin{bmatrix} app & apq \\ aqp & aqq \end{bmatrix}$$

Let p be the row number and *q* be the column number of the the pivot element for Jacobi transformation. Then θ, the angle between -45^0 to 45^0 through which rotation operation is to be performed, is given by,

$$\tan(2\theta) = \frac{a_{pq} + a_{qp}}{a_{qq} - a_{pp}} \qquad \theta \in \{-\Pi/4, \Pi/4\}$$

The Jacobi matrix, which is nothing but the rotational operator, is then constructed as ,

$$J = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

Applying Eq. (3), the iterative equation is given by,

$$A^{i+1} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}^{T} \begin{bmatrix} app & apq \\ aqp & aqq \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

$$J = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & c & 0 & 0 & s & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -s & 0 & 0 & c & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

.

Figure 2. Jacobi Transformation of nxn Matrix Selection of Pivotal Elements

For a matrix *A* of size nxn as per Eq.(3), the Jacobi transformation is applied iteratively, computing Jacobi matrix in each iteration from the previously iterated matrix *A* with pivot values *p* and *q* . This freezes the four elements *app, apq, aqp* and *aqq* of a matrix *A* leading to computation of and hence cos c) and sin s values. This will generate the Jacobi matrix with elements *J(p,p), J(p,q), J(q,p), J(q,q)* as *c, s, -s, c* respectively and remaining diagonal elements being forced to 1 while non-diagonal elements being forced to zero. Consider an example of 8x8 Jacobi matrix with $p = 2$ and $q = 5$ where except for the four elements as above, all diagonal elements are 1 while non-diagonal elements are 0.

3.3 Singular Value Decomposition for a n x n Matrix

Once Jacobi matrix is generated from A^i (matrix *A* in the ith iteration), $A^{(i+1)}$ is computed using Jacobi transformation $J^T A^i J$. The process is repeated till the computed *A* matrix has non-diagonal elements less than some threshold value (hence forcing them to zero) and diagonal elements are nonzero. The diagonal elements are then arranged in a descending order. The square root of these diagonal elements gives a set of singular values for the original matrix A. This algorithm of SVD is represented in Figure 3.

Selection of pivot plays a major roll in reducing the time complexity. The basic Classical Jacobi method by Brent [1963] which has a time complexity of $O(n^3)$[2] uses position of maximum value in a row as pivot position. The cyclic Jacobi method[1966] is a modification of classical Jacobi method wherein the starting pivot element corresponds to $p = 1$ and $q = 2$ (i.e. $p + 1$) with p incrementing by 1 in each successive iteration. This selection method provides the complexity of $O[(n(n-1)/2)]$ [3].

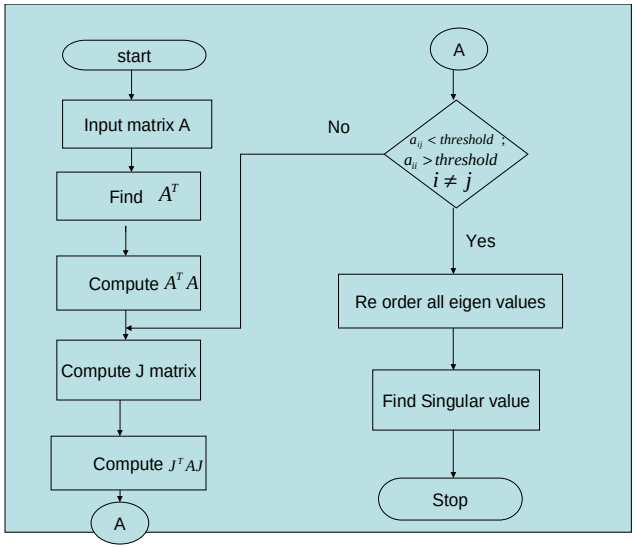

Figure 3. Flow Chart for SVD Computation

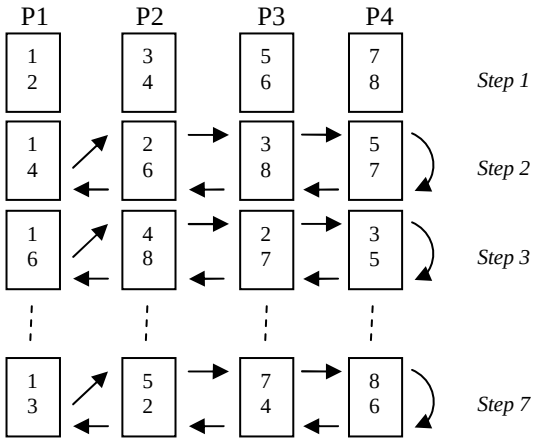

Figure 4. Tournament Ordering Scheme for 8×8 Matrix

In 1985 Brent and Luk introduced a more efficient way of determining the pivotal element called Tournament Ordering[10] which has a time complexity of *O(2(n-1))*. An example of pivoting sequence where the four processors P_1, P_2, P_3 and P_4 are working in parallel on diagonal for a 8x8 size matrix is shown in Figure 4. In step1, the four processors generate Jacobi matrix by considering the pivot with p and q values as shown in the first row of the following figure. In the subsequent steps the processors work by shifting the pivot p, q values in a fashion as shown by the arrows in Figure 4.

We have implemented all the three methods using MATLAB before implementing them in FPGA. Table 1 gives MATLAB simulation results of pivot selection on an image of size (40x40). It can be concluded from this table that tournament ordering scheme is the best among the three selection methods mentioned above and can be taken further for FPGA implementation. Also it is obvious from the table that the number of iterations required to obtain a desired error is consistent with the time complexity of each method as mentioned above.

Table 1. Matlab Simulation Results (Pivot Selection Methods)

Method	No of Iterations	% error	Convergence found	Comments
Classical Jacobi	20000	1.273	No	Error does not reduce below 1.27% even upto 60000 iterations
Cyclic Jacobi	5000	0.004	Yes	Error does not reduce below 0.004% even upto 8000 iterations
Tourna-ment Ordering	100	0.001	Yes	Considerably lower no of iterations to obtain 0.001 % error. Best pivit selection method

4. FPGA IMPLEMENTATION

Figure 5 shows the block diagram of the complete system consisting of four stages viz. image acquisition, image segmentation, feature extraction and classification. Out of these first two stages are implemented in host PC while the remaining two stages are implemented in FPGA. The emphasis of this paper is on FPGA implementation of SVD using Jacobi method with tournament ordering for feature extraction and classification of iris templates. The segmented iris template image which has a size of 40x40 pixels with 8-bit pixel data (256 gray levels) is sent via parallel communication port to FPGA kit for further processing. The next sub-section gives the details of SVD hardware architecture while Sub-section 4.2 presents the details on classification of iris images represented by SVD features using one of the classifiers such as Hamming Distance Classifier.

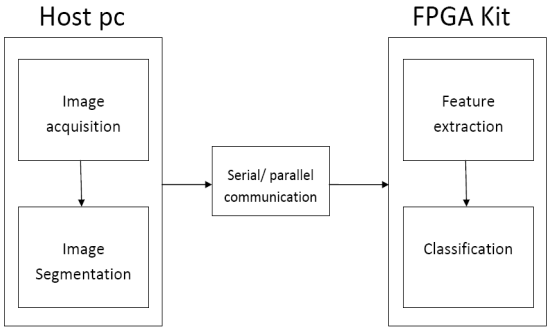

Figure 5. Block Diagram of the System

4.1 SVD Core

Figure 6 shows SVD core for FPGA implementation. The major components of SVD core are Matrix Multiplier, Jacobi Transformer, Finite State Machine, Reorder Unit and Square Root block.

Matrix multiplier is used for computing one of the three matrix multiplications viz. A^TA, J^TA and J^TAJ. The appropriate input matrix to this unit is selected using Matrix Multiplexer which selects original image matrix for the first step, selects J^T for the second step and C and J for the third step. Matrix multiplier based on systolic array parallel implementation gives result for n×n size matrix in n clock cycles. Matrix multiplier has extra facility for getting results in normal format or Q15 format and perform transpose multiplication (e.g. J^TA). The outputs of this unit are multiplexed to Jacobi transformer and Matrix

Multiplexer or Reorder Matrix block depending upon the convergence of the matrix *A* after Jacobi transformation.

Jacobi Transformer generates Jacobi matrix by operating on *A* and selecting pivot element by tournament ordering method. This block contains three sub blocks viz. diagonal processor, sine-cosine look up table and update pivot. Diagonal processor requires pivot elements as input and provides sine and cosine of the angle computed using look-up table in Q15 format. Update pivot block generates new pair of pivot elements for construction of new Jacobi matrix J as per scheme given in Figure 4.

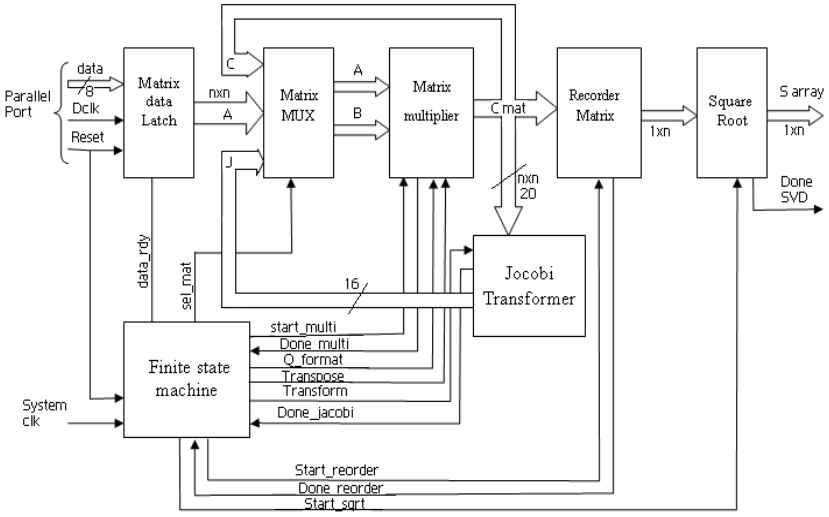

Figure 6. SVD Core for FPGA Implementation

Finite State Machine is the heart of SVD core which generates the control signals as per the state diagram shown in Figure 7. The various operations carried out during each state are also indicated on the state diagram. Reorder Matrix block implements sequential search algorithm to arrange the diagonal elements in descending order while square root block implements non-restoring type method to compute square root of diagonal elements which are primary outputs of SVD core.

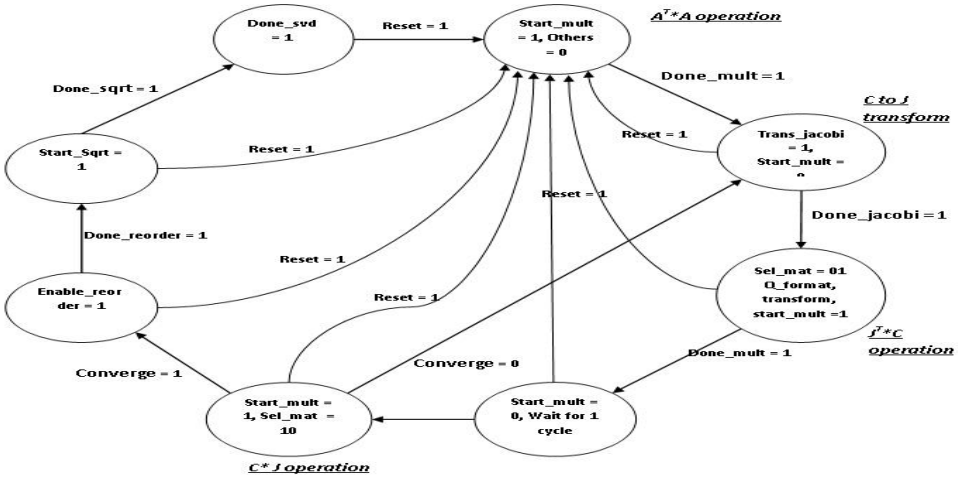

Figure 7. State Diagram of the Control Unit (Finite State Machine)

The operation of SVD core is as follows:

Matrix Data Latch has data input from PC provided with data clock and asynchronous reset. Reset from host PC resets the whole system and finite state machine. Valid data is latched at the rising edge of the data clock and stored in the matrix form. This latch generates *data ready* signal after receiving the whole matrix data and triggers FSM for the next state. FSM generates the *sel_mat* signal to select either the original

matrix *(A)* for conversion to square matrix in the first step, or Jacobi Transpose/Jacobi matrix (J) in the second/third step or J and C matrix (which is the result of $J^T A$) in the third step. The matrix multiplication is initiated by the *Start Multiplication signal* generated by FSM while *done* multiplication signal from Matrix Multiplier is used to trigger FSM for executing the next Jacobi Transformer operation. It may be noted that with large number of Jacobi matrix elements being zero, the 16x16 multiplier output is limited to only 20 least significant bits. Jacobi transformer block constructs Jacobi matrix after computing θ value for the pivot *p, q* values as per tournament ordering scheme and then refers $\cos\theta$ and $\sin\theta$ values for the computed θ from look-up table by generating appropriate address. Since $\cos\theta$ and $\sin\theta$ values do not change appreciably for θ below -10^0 and above $+10^0$, the look-up table entries in Jacobi transform block are also limited to $\pm 10^0$ with 1024 entries in ROM with 10 bit address giving 0.008^0 resolution. *Done* signal from Jacobi Transformer triggers FSM either to repeat the next iteration or enable Reorder Matrix to accept the diagonal values of the converged *A* matrix. The handshaking between Reorder Matrix, FSM and Square Root block initiates square root operation to find out singular values and then to output those with *Done SVD* signal being generated by the same block.

Hamming Classifier

Classification is the last stage of iris recognition. An architecture of this is as shown in Figure 8. A set of features obtained from SVD are stored in ROM for all iris images under consideration. SVD values for the iris image to be recognized, as computed by SVD core, are inputted to the classifier *(S-array)* which computes simple hamming distance between SVD values of the test image with pre-stored values of all images one by one. The classifier outputs 1-hot encoded value of the image if the difference between corresponding singular values for all singular values is less than pre-determined threshold value. Thresholding level can be decided by finding out tradeoff between false acceptance ratio and false rejection ratio.

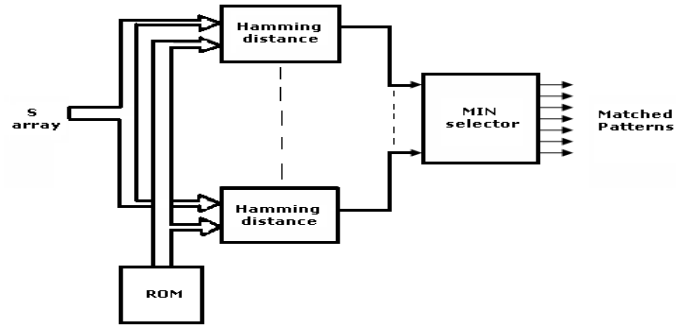

Figure 8. Hamming Classifier for FPGA

5. SIMULATION RESULTS (XILINX)

Both SVD core and Hamming Classifier were implemented in Vertex 5 XC50VXL FPGA device using Xilinx ISE 10.1 EDA tool. SVD results for few iris images as computed by MATLAB (using floating point arithmetic) and by FPGA implementation (using fixed point arithmetic) is as shown in Table 2.

Table 2. Comparison between SVD Values

Image No.	SVD Values Computed using MATLAB				FPGA			
	S1	S2	S3	S4	S1	S2	S3	S4
1	409	16	8	5	408	13	11	9
2	427	69	31	7	423	70	38	13
3	589	16	9	3	588	14	9	6
4	619	10	5	2	614	15	7	4
5	463	13	11	5	463	14	7	5

It is seen from the above table that there is a close match between SVD values computed using MATLAB – a software approach and using FPGA implementation – a hardware approach. Further it was found that the Hamming classifier (with ROM storing MATLAB computed SVD values for ten images)

identified the test image correctly the image data of which was fed to SVD core. These results validate the hardware architecture for both SVD core and the classifier.

The expected time complexity of various blocks of FPGA implemented architecture for an image of size *nxn* is given in Table 3. The synthesis report of this architecture as generated by XST tool of Xilinx ISE for an iris image of size 40x40, when operated with a clock of 100 MHz indicated that the time consuming block is a SVD core with 16 μs (1600 pixels x 10 ns) as the time for capturing the image data and 203 μs as the time for SVD computation. Hamming Classifier takes negligible time in comparison with this. This result indicates that the iris image can be identified in a time period less than 250 μs which is much faster than that identified by a human eye. This further suggests to reduce the clock frequency suitably to reduce the overall power consumption in the hardware. The same synthesis report indicated the resource utilization of about 80% suggesting the suitability of the FPGA device used.

Table 3. Xilinx Timing Simulation Results

Block name		Clock Cycles
Matrix data latch		n^2+1
Matrix multiplexer		1
Matrix Multiplier		n
Jacobi Transformer	Pivot update	1
	Read lookup	1
	Division 32 by 20	n+1
	Total	n+3
Finite State Machine	5 states 1 clock cycle for each	5
Convergence		1
Reorder		n
Square root		n

6. CONCLUSION

A hardware architecture for computing SVD values of a given 2-D matrix is developed and successfully implemented in Virtex5 FPGA of Xilinx. This has been used to extract a set of features (<n) for a nxn iris image data. This is the greatest advantage of SVD algorithm from data storage point of view. A close match between SVD values obtained using MATLAB and FPGA implementation validates the architecture developed. A Hamming Classifier designed using a set of comparators with different reference values successfully classified the test iris images on sample basis. The timing simulation indicated that the core can identify the iris images successfully even when operated with a clock of 10 MHz.

7. FUTURE SCOPE

Though the hardware developed and implemented resulted into successful iris image identification, the experimentation was performed on few benchmark images. There is a need to validate the hardware further by checking the False Acceptance Ratio (FAR) and False Rejection Ratio (FRR) on a large data base. Hardware optimization is another area where special attention can be given to reduce resource utilization and delays. In this aspect, hardware algorithms for implementing division and square root demands further research. Studies can be carried out to find out the power dissipation and implement the low power approaches. Fixed point arithmetic results can be made more accurate with increasing Q-value in Q-format and increasing the number of lookup table entries, while considering the trade-off between area and accuracy.

ACKNOWLEDGEMENTS

The authors would like to acknowledge Institute of Automation, Chinese Academy of Sciences for making CASIA iris image database available on the net which has been extensively used for carrying out the research work.

REFERENCES

[1] Brent R. P., and Luk, F. T., " An Efficient Jacobi-like Algorithm for Parallel Eigen value Computation", *IEEE transactions on computers*, vol. 42, no. 9, September 1993, pp 1058-1065.

[2] Rahmati, Sadri, Naeini, "FPGA based singular value decomposition for image processing applications" Proc, of *International Conference on Digital Object Identifier, 2008*, pp 185-190.

[3] Weiwei, M. E. Kaye, D. M. Luke, R. Doraiswami, "An FPGA-Based Singular Value Decomposition", Proc. Of *Canadian Conference on Digital Object Identifier*, 2006, pp 1047-1050.

[4] Nikolay Sorokin, "Implementation of high-speed fixed-point dividers on FPGA", *Journal of CS and T* , April 2006, Vol. 6, pp 8-11.

[5] David C. O'Neal and Raghurama Reddy "Solving Symmetric Eigenvalue Problems On Distributed Memory Machines" Pittsburgh Supercomputing Center Carnegie Mellon University Pittsburgh, PA 15213, October, 1994, pp 1-27.

[6] K piromsopha, "An FPGA implementation of fixed point square root operation", *ISCIT 2001*

[7] Liu-Jimenez, Sanchez-Reillo, Sanchez-Avila, "Full hardware solution for processing iris biometrics", Security Technology, 2005, *CCST '05*.

[8] Ignacio Bravo, Pedro Jimnez, Manuel Mazo,Jos Lzaro, Alfredo Gardel, "Implementation of Jacobi Method to Solve the eigen value and eigen vector problem ",

[9] R.P.Brent and F.T.Luk, "The solution of singular value and symmetric eigen value problems on multiprocessor Array", *SIAM J.sci Stat.Comput*,Vol.6,No.1,pp.69-84 Jan1985

[10] Michael W. Berry, Dani Mezher, Bernard Philippe, and Ahmed Sameh "Parallel Algorithms for the Singular Value Decomposition" *Handbook on Parallel Computing and Statistics*, 2005, pp 117-164

[11] Li Ma, Tienui Tan, Fellow,IEEE,Yunhong Wang,Member,IEEE,and Dexin Zhang "Efficient Iris Recognition by Charactering Key Local Variation", *IEEE TTranscations on Image Processing*, vol 13, 6, June 2004,

[12] O.Dniz,M.Castriiin,M.Hernndez, "Face recognition using independent component analysis and support vector machines, pattern Recognition Letters"Vol.24,No13,pp.2153-2157,Sep.2003.

BIBLIOGRAPHY OF AUTHORS

Mr. Babasaheb G. Patil :He received his M.E. Electronics degree in 1990 and B.E. Electronics in 1988. He is currently working as a Associate Professor in department of Electronics in Walchand College of Engineering, Sangli, Maharashtra, India. He is having keen interest in image processing and communication. Currently he is carrying out research work in the field of Image Processing.

Dr. (Mrs) Shaila Subbaraman : She received M-Tech degree from IISc. Bangalore in 1975 and Ph.D. from IIT Bombay in 1999. She worked in Semiconductor Device Manufacturing company from 1975 to 1989. Currently she is Professor in Department of Electronics in Walchand College of Engineering, Sangli, Maharashtra, India. She has keen interest in the field of Microelectronics and VLSI Design.

Mr. Nikhil Niwas Mane: He is a M.Tech. degree from Walchand College of Engineering, Sangli, Maharashtra, India in Electronics. He has 2 years industrial work experience. Presently he is an Assistant Professor in Bharati Vidyapeeth's College of Engineering, Kolhapur, Maharashtra. He has keen interest in the field of embedded system and VLSI Design.