International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 11, November - 2013

Network on Chip Using GMM Based Image Classifier

[1] P.ManigandaM.E.(Ph.D),Asst.Prof [2] M.Phanikanth.M.tech,Asst.Prof [3] P.Ganapathi.M.Tech,Asst.Prof

[1,2,3]C.V.S college of Engg, Tirupati. INDIA

Abstract

The aim of this paper is to give briefing of the concept of network-on-chip and programming model to **Gaussian mixture models (GMM)-based classifiers have shown increased attention in many pattern recognition applications. Improved performances have been demonstrated in many applications, In this paper, first the performance of GMM and its hardware complexity are analyzed and compared with a number of benchmark algorithms. First, a serial-parallel vector-matrix multiplier combined with an efficient pipelining technique is used.\ A novel exponential calculation circuit based on a linear piecewise approximation is proposed to reduce hardware complexity. The precision requirement of the GMM parameters in our classifier are also studied for various classification problems. The proposed hardware implementation features programmability and flexibility offering the Noc in possibility to use the proposed architecture for different applications with different topologies and precision requirements.**

Index Terms: *Network On Chip*, GMM, pattern recognition, reconfigurable architecture.

I. Introduction

Network-on-Chip (NoC) is an approach to design the communication subsystem between IP cores in a System On a Chip (SoC). NoCs can span synchronous and asynchronous clock domains or use unclocked asynchronous logic. This NoC brings an effective improvement over conventional busses and cross bar switches. The power requirement of the SoC is high where as it can be reduced by the NoC architecture. NoC is an developing paper in the field of VLSI. Since the use of emerging NoC architecture in VLSI it reduces the size of the architecture due to the reduced amount of buses and transmission lines. The GAUSSIAN mixture models (GMM) classifier has gained increasing attention in the pattern recognition community. GMM can be classified as a semi-parametric density estimation method since it defines a very general class of functional forms for the density model. In this mixture model, a probability density function is expressed as a linear combination of basis functions . While GMM provides very good performances and interesting properties as a classifier, it presents some problems that may limit its practical use in real-time applications. One problem is that GMM can require large amounts of memory to store various coefficients and can require complex computations mainly involving exponential calculations. Thus, this scheme can be put to efficient practical use in Network on Chip implementation strategies are developed. In this paper, we propose Network On Chip GMM-based classifiers. First, after analyzing the complexity of the GMM classifier it was found that the vector-matrix multiplication and the exponential calculations are the most critical operations in the classifier. A good tradeoff between real-time processing and hardware resources requirements is obtained using a serial- parallel architecture and an efficient pipelining strategy. Second, a linear piecewise function (LPF) is proposed to replace the exponential calculation.. The effect of both limited precision and the mixture models approximation using LPF on the classification performance is investigated using seven different data-sets. These data-sets are also used to compare the performance of GMM with other benchmark classifiers.The design was made flexible and programmable making it a general purpose processor which can be applied to different classification problems.

II.Hard ware Proposal

Noc components:

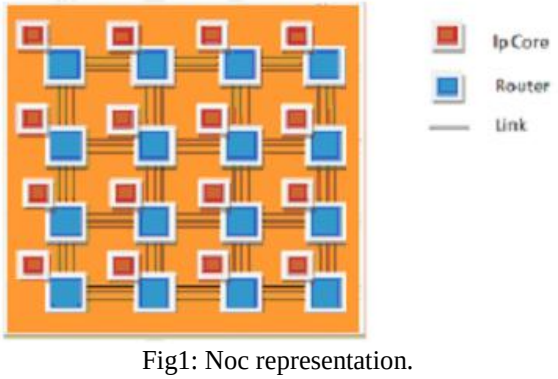


Fig1: Noc representation.

Network Interface Controller (NIC). The NIC implements the interface between each IP node and the communication infrastructure. The architecture of the NIC component can be divided into two modules. The first one is focused on the interaction with the computation or memory node bus, and the other one is focused on the interaction with the rest of the NoC. This component is called in many different ways in NoC literature: NI for Network Interface, NA for Network Adapter, or RNI for Resource Network Interface are some examples.

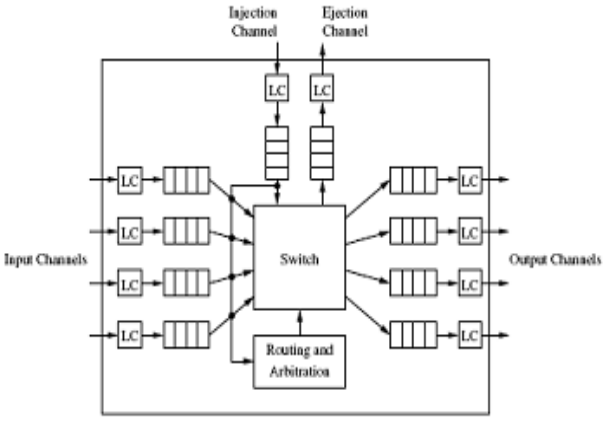
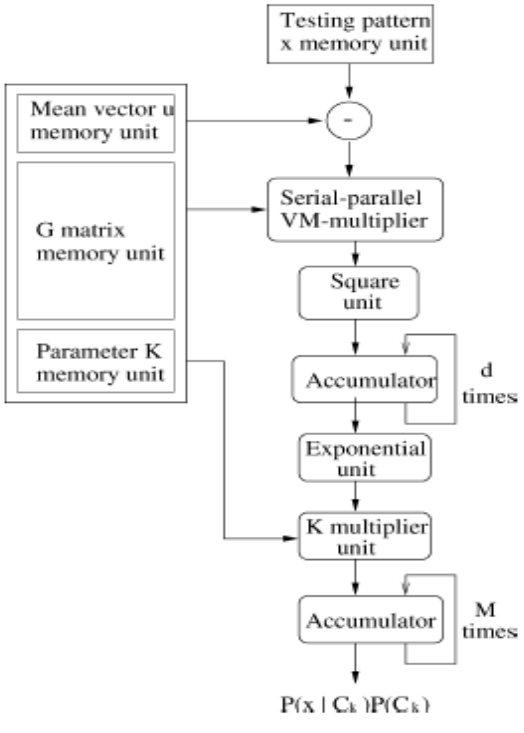


Fig 2:Generic Router block diagram

Router. Also called switch. These components are in charge of forwarding data to the next tail. On the routers we can find implemented the routing protocol, buffer capabilities and the switching method. In general, the router component is composed of the next elements: Arbitrer, which its main task is to grant channels (selecting an input port and an output port) and route packets; Crossbar, of n input x n output ports that direct the input packet to the corresponding output port; and buffer or queue, if that is the case – as it is in the packet switching protocols – which is used to buffer incoming and outgoing data in the router.

2.Data Flow Blocks Of a GMM Classifier:



i. Serial-parallel Vector-matrix Multiplier.
ii. Linear Piecewise Function Unit.
iii. WTA (Winner-takes-all) Circuit.

4.Functional Blocks (Algorithm)Of The GMM Classifier System:

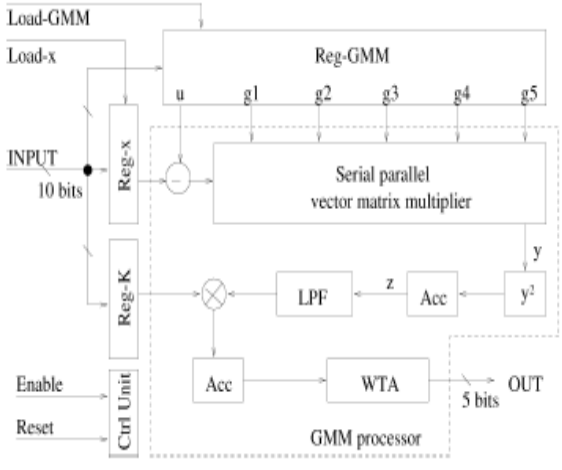


Fig4:the functional block diagram of the overall GMM classifier system.

4.1.Reg-GMM:

The architecture includes two main registers *Reg-X* and *Reg-GMM* used to store the input pattern **x** and the GMM parameters (μ,G, k) respectively. The main computational block of the system is the GMM processor used to calculate $\delta(x/C_k)$ $\delta(C_k)$ for each class in turn and, hence, makes the final class assignment decision for a given input pattern.

4.2.Serial Parallel Vector Multiplexer:

The GMM processor includes a serial parallel vector matrix multiplier, a square and multiplier units, a LPF unit, a winner takes- all (WTA) circuit, and two accumulators. A 10-bit data bus is used in order to load the GMM parameters and the test vector data using the load signals .Once all the parameters are loaded, the GMM processor can operate on real-time basis by processing data in a systolic fashion. Initially, the $\mathbf{G_j}$ matrix is multiplied by the $\mathbf{S_j = x-\mu_j}$ vector. The resulting vector Y_j is then fed to a square unit followed by an accumulator which performs the summation of all the squared z_j components of the vector resulting in the value z_j as described in

4.3.LPF unit:

The result is fed to the LPF unit and is multiplied by constant K_j The multiplication result represents a single parameter $K_j \exp\{-Z_j\}$, which when accumulated **M** times will lead to the value of $\delta(xC_k)\ \delta(C_k)$ as described by (15). An accumulator is, therefore, required at the output of the exponential block which is iterated M times.

4.4..WTA CIRCUIT:

The values $\delta(xC_k)\ \delta(C_k)$ are then compared one by one and the maximum value is selected and its corresponding

class is assigned to the test pattern. In what follows, we will describe the important building blocks of the GMM processor including the serial parallel vector matrix multiplier, the LPF unit as well as the WTA circuit.

5. Serial-parallel Vector-matrix Multiplier:
Vector triangular matrix multiplication for d=3

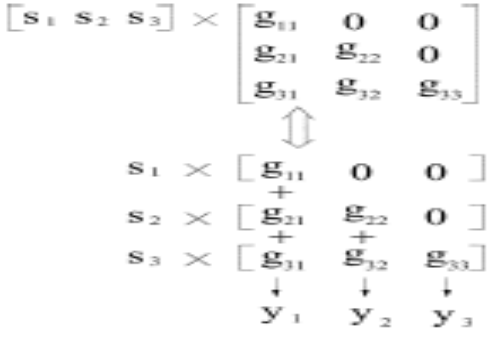


Fig5:Vector triangle multiplication for d=3.

where y1=s1g11+s2g21+s3g31.
y2=s2g22+s3g32.
y3=s3g33.

Where $\mathbf{S_j = x - \mu}$ The vector-matrix multiplier is used to calculate , $Y_j = S_j^T G_j$ where $\mathbf{S_j = x - \mu}$ The vector-matrix multiplier is optimized for our specific need of vector-triangular matrix multiplication. Fig. 8 describes the vector-triangular matrix multiplication for d=3

The multiplication can be simplified by decomposing the **G** matrix into row vectors which are then multiplied by components. Once this is achieved, the final resulting row vector **Y** is obtained by summing-up the intermediate results row wise. One can note that $\mathbf{Y_3}$ just requires one multiplication whileY_2 and $\mathbf{Y_1}$ require two and three multiplications, respectively.

Due to this property, we can first multiply $\mathbf{S_3}$ with the 3rd row of G to generateY_3 and the partial results of Y_1 and $Y_{2\ are}$ temporarily accumulated. Next, S_2 is multiplied with the 2nd row of G and Y_2 can be generated after accumulation. This multiplication was achieved using an efficient systolic architecture illustrated in Fig.5, which permits to obtain a good tradeoff between operation speed and chip area. The elements of vector **S** are fed into the multiplier serially while the row components **G** of are provided in parallel.

5.1.Serial parallel vector-matrix multiplier:

For simplicity reasons, the figure only shows a maximum dimension of 5. Note that it is possible to use the structure for a lower dimension since zero components can be inserted in non utilized blocks. The vector matrix multiplier is used to calculate $Y_j = S_j^T G_j$ where $\mathbf{S_j = x - \mu}$ The multiplication can be simplified by decomposing the 'G' matrix into row vectors which are then multiplied by 'Sj' components. Once achieved the final resulting row vector 'y' is obtained by summing-up the intermediate results row wise

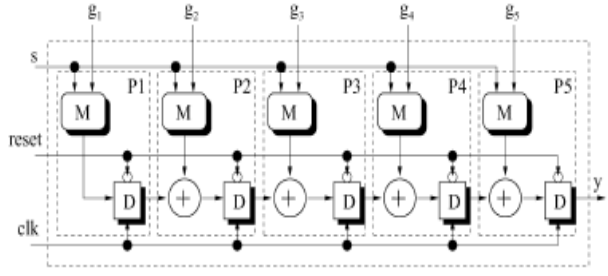


Fig6:Serial-Parllel vector-matrix multiplier.

5.2.Computation Sequences Diagram For Serial-parallel Vector Multiplier:

*Serial Parallel Vector Matrix Multiplier Combined with an Efficient Pipelining Technique is used.

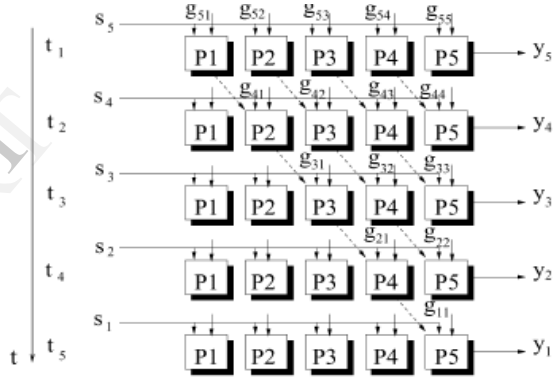


Fig7:This computation sequence shows five clock cycles**.**

y1=s1g15+s2g24+s3g33+s4g42+s5g51.
y2=s2g25+s3g34+s4g43+s5g52.
y3=s3g35+s4g44+s5g53.
y4=s4g45+s5g54.
y5=s5g55.

At the first clock cycle (t_1), g_{5i} are fed to the vector matrix multiplier together with S_5. During this first cycle, output $\mathbf{Y_5 = S_5 g_{55}}$ is obtained. At the next clock cycle (t_2),g_{4i}, are fed to the vector matrix multiplier together with S_4 and $\mathbf{Y_4 = s_4 g_{54} + s_4 g_{44}}$ is obtained. The procedure is continued until the resulting vector components y_i are obtained in five clock cycles.

6.Linear piecewise function unit:

*A novel exponential calculation circuit based on linear piecewise approximation is proposed to reduce hardware complexity.

*The proposed hardware implementation features programmability and flexibility offering the possibility to use the proposed architecture for different applications with different topologies.

* The linear piecewise function unit; the circuit can be configured in three different modes realizing i.e. f1 (z), f2 (z), f3 (z).

6.1.The Architecture of the Linear Piece Wise Function Unit:

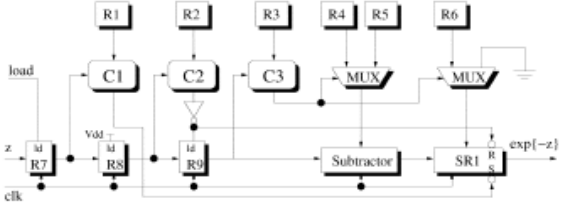


Fig8:Architecture of Linear piecewise Function

1. In this digital architecture of LPF unit requires R1 to R6 registers to store the different parameter of approximation.
2. R7 is input register which is loaded by the input data z (40 bits)
3. SR1 is 40-bit shift register with set and reset options, SR1can shift the data by a number of bit set by the value stored in R6.
4. Three comparators C1 to C3 is allowing to compare the input data z with the values stored in registers R1 toR3.
5. The LPF unit operates in three possible linear piece wise functions i.e. f1 (z), f2 (z) and f3 (z).

MODE 1:

$$f_1(z) = \begin{cases} 1, & \text{if } z < a \\ 0, & \text{if } z \geq a \end{cases}$$

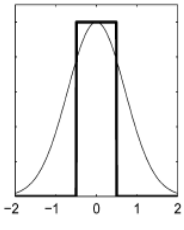
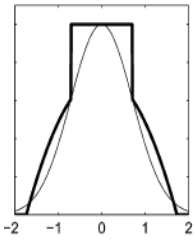


Fig9:The exponential approximation of f1 (z) on Gaussian model

The implementation of f1(z) is straightforward.
A comparator can be used to compare **z** and **a** If **z<a** the output is 1, otherwise, the output is 0. For $f_2(z)$ two comparators are required. If z<a or z ≥b, the output is 1 or 0
•In the f1 (z): 'a' is stored in both R1 and R2, C1 compares input z with a.
•If z<a, the output of C1 is low and the output of SR1 will be set to 1.
•If z>=a, the output of C2 is high and this signal is used to reset SR1.

MODE 2



$$f_2(z) = \begin{cases} 1, & if \quad z < a \\ (b-z), & if \ a \leq z < b \\ 0, & if \quad z \geq b \end{cases}$$

Fig10: the exponential approximation of f2(z) on Gaussian model

•In the f2(z): 'a' is stored in R1 and R3 while 'b' is stored in R2 and R5.
•C1 operates similarly to the case described in mode 1.
•If z>=a, C2 will compare z and b SR1 will be reset,.
•z will be loaded to register R9 and C3 will compare z with content of R3.
•(b-z) is then calculated and loaded into SR1(C3 high).
•The output of SR1 is (b-z) if a<=z<b.

MODE 3:

$$f_3(z) = \begin{cases} 1, & if \quad z < a \\ 2^{n-m}(b'-z), & if \ a \leq z < b \\ (c-z), & if \ b \leq z < c \\ 0, & if \quad z \geq c \end{cases}$$

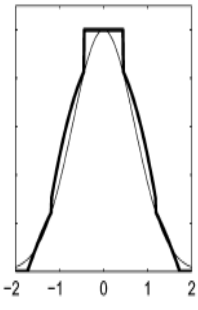


Fig11:The exponential function approximation of f3 (z) on Gaussian model

•In the f3 (z): **'a', 'b', 'b`'** are stored in R1, R3 and R4, respectively, while 'c' is Stored in R2 and R5. R6 stores the value of (n-m).
•If z<a, C1 will set SR1, later will be reset by C2 if z>=b.
•In the case where b<=z<c, the operation is the same as that of mode2 and the output of SR1 will be (c-z).
•If a<=z<b, the output of C3 will be low and the subtract or will calculate (δ_i - s)
•The result is shifted to the left by (n-m), the output of SR1is set to 2^{n-m}(b` - z)

6.WTA CIRCUIT

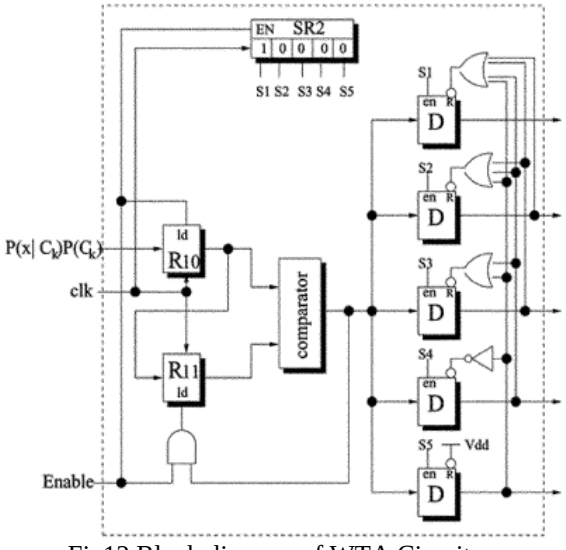


Fig12.Block diagram of WTA Circuit.

•In WTA circuit, the values $\delta(x/C_k)$ $\delta(C_k)$ need to be compared one by one before making the final class assignment decision. The maximum value is selected and its corresponding class is assigned to the test pattern.
•Initially R10 AND R11are reset and the initial control sequence stored in the shift register SR2 is "10000".
•Each bit within the 5-bit control sequence stored in SR2 is responsible for enabling one D-FF, by shifting control sequence in SR2 the output of the comparator is loaded in a new D-FF and comparing component
•The result stored in D1, similarly D1 to D5 will be zeroed except for one FF which will corresponds to

III.Result:

1.Out put wave form for GMM Classifier:

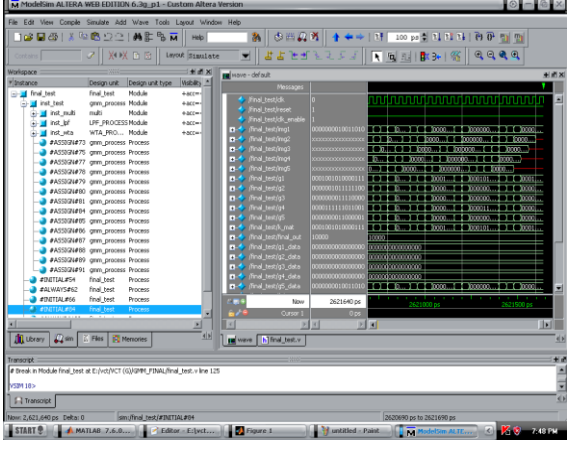


Fig13:Output waveform of GMM Classifier.

2.Input Image:

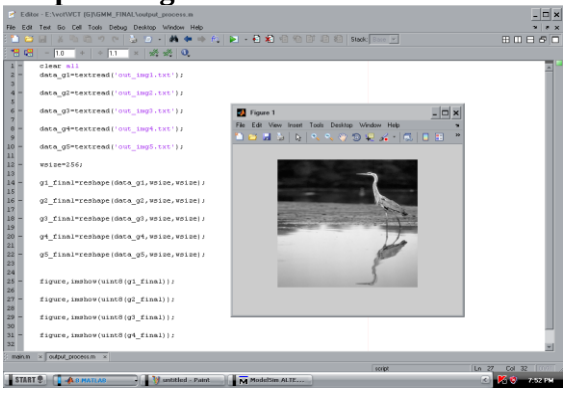


Fig13:Input Image for GMM Classifier.

3.1.Output Images for GMM Classifier:

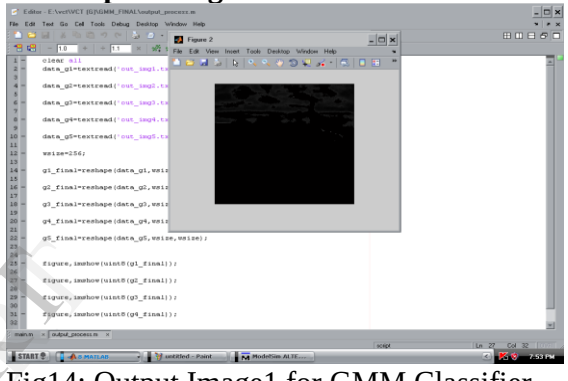


Fig14: Output Image1 for GMM Classifier

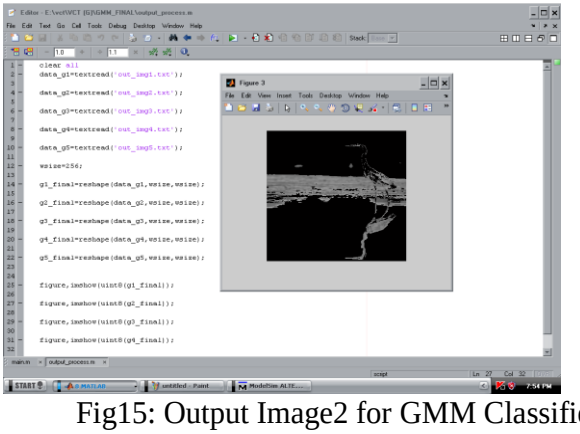


Fig15: Output Image2 for GMM Classifier

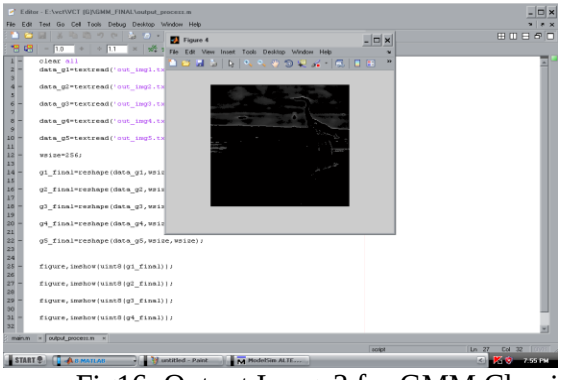


Fig16: Output Image3 for GMM Classifier

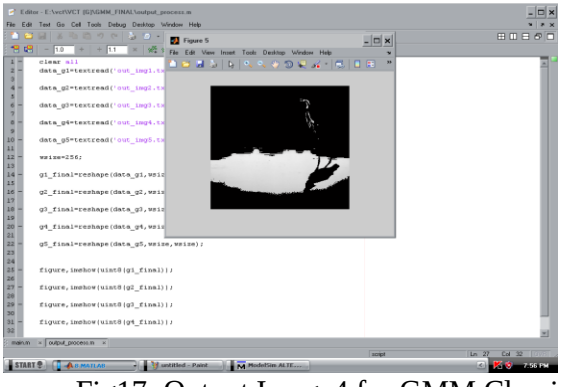


Fig17: Output Image4 for GMM Classifier

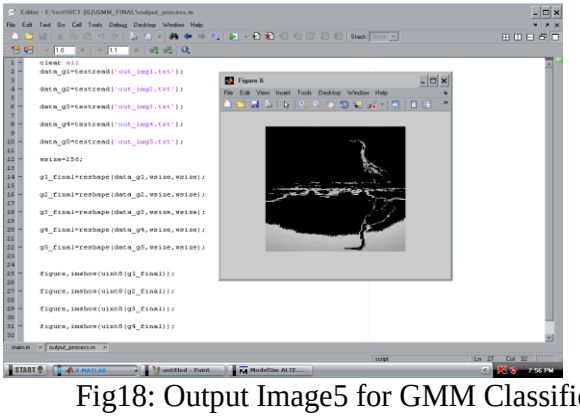


Fig18: Output Image5 for GMM Classifier

2.Software's Used:

- **Xilinx**
- **Mat Lab**

3.Applications:

*G.M.M. Based classifiers are used to pattern recognition application.
*Classifying the data (image**).**
*Medical applications.

4.Advantages:

*Classification rate is high.
*As comparing this is accurate method of result.

IV. CONCLUSION

In this paper, Network-on-chip topic remains as an attractive research field for academia that raises its popularity year by year .we presented a pattern recognition system based on a GMM classifier. Simulation results suggest that the GMM classifier presents the best classification performance with acceptable complexity when compared to KNN, MLP, RBF, and PPCA. This hardware-friendly architecture is based on the use of power-of-two coefficients as well as LPF-based GMM. The proposed procedure consists of building an original GMM based on the training data-set and then optimizing the parameters of the LPF-based GMM using the classification performance as an optimization criteria

A prototype chip was designed using automatic placement and routing based on 0.25- m technology occupying an area of 1.69 mm . Successful operation of the architecture is demonstrated using NOC through simulation results as well as experimental tests for gas identification application requiring ten Gaussian models and five classes. A classification performance of 92% was achieved for 100 input patterns processed in less than 57 s. To the best of our knowledge, this prototype represents the first reported hardware implementation of a GMM classifier.

V.REFERENCES

[1]D.Castells-Rufas, J. Joven, S. Risueño, E. Fernandez, J. Carrabina, T. William, H. Mix. "MPSoC Performance Analysis with Virtual Prototyping Platforms". PSTI, San Diego, USA, September, 2010

[2]J. Joven, "A lightweight MPI-based programming model and its HW support for NoC-based MPSoCs", PhD Forum DATE, IEEE/ACM Design, Automation and Test in Europe (DATE'09), April 2009, Nice, France.

[3]Hilton, C.; Nelson, B.; , "PNoC: a flexible circuit-switched NoC for FPGA-based systems," Computers and Digital Techniques, IEE Proceedings - , vol.153, no.3, pp. 181-188, 2 May 2006

[4]Pavlidis, V.F.; Friedman, E.G.; , "3-D Topologies for Networks-on-Chip," Very Large Scale Integration (VLSI) Systems, IEEE Transactions on , vol.15, no.10, pp.1081-1090, Oct. 2007

[5] S. L. Phung, A. Bouzerdoum, and D. Chai, "Skin segmetation using color pixel classi fication: Analysis and comparision," IEEE Trans. Pat- tern Anal. Mach. Intell. , vol. 27, no. 1, pp. 148 –154, Jan. 2005.

[6] S. B-Belhouari, A. Bermak, M. Shi, and P. Chan, "Fast and robust gas identification system using an integrated gas sensor technology and Gaussian mixture models," IEEE Sensors J. , vol. 5, pp. 1433–1444,Oct. 2005.

[7] Y. Huang, K. B. Englehart, B. Hudgins, and A. D. C. Chan, "Opti- mized Gaussian mixture models for upper limb motion classification, "in Proc. 26th Ann. Int. Conf. Eng. Med. Bio. Society, (EMBC), Conf.2004, pp. 72–75.
[8] C. P. Lim, S. S. Quek, and K. K. Peh, "Application of the Gaussianmixture model to drug dissolution profiles prediction, " Neur. Comput.Appl., vol. 14, pp. 345 –352, 2005.

ACKNOWLEDGEMENT

Pujali.Maniganda, M.E (PhD),
Associate.Prof. in Chadalawada Venkata Subbaiah College of Engineering in Tirupathi. & Perusing Ph.D in Veltech Dr.RR. & Dr.SR. Technological University in Chennai.;

M.Phanikanth M.Tech
Asst..Prof. in Chadalawada Venkata Subbaiah College of Engineering in Tirupathi.

P.Ganapathi M.Tech
Asst..Prof. in Chadalawada Venkata Subbaiah College of Engineering in Tirupathi.