Proc. of Int. Conf. on Current Trends in Eng., Science and Technology, ICCTEST

Speed Forgy's K-Means Clustering Algorithm with Single Clock Cycle Divider for Image Analysis.

Anuradha.M.G[1] and Dr. Basavaraj.L[2]
[1]Research scholar, ATME college of Engineering, Department of ECE, Mysore
Email: anuarun.19@gmail.com
[2]Principal, ATME college of Engineering,Mysore
Email: principal@atme.in

***Abstract*— In this paper K-Means clustering algorithm has been implemented. The proposed K-Means algorithm architecture is adaptable to the feature vectors with different dimensions to effectively utilize the system resources. Manhattan distance calculators which do not require multipliers are being used in the design to optimize the performance of the hardware. The division module realized for finding the new centroid in the architecture carries out the division operation in a single clock cycle, hence each iteration for finding a new centroid requires only one clock cycle irrespective of the number of input vectors / input vector dimensions. This modification to division operation reduces the time of clustering operation. The K-Means hardware is checked for its functionality using Modelsim simulator and is synthesized using Cadence RTL Compiler with 180-nm CMOS technology, and the experimental results show that the gate count and the maximum frequency are 600K and 300 MHz, respectively.**

***Index Terms*— Unsupervised Clustering, Forgy's K-means, hardware architecture.**

I. INTRODUCTION

Image analysis is extraction of meaningful information from an image that unfolds the underlying information. To understand the underlying information of a video program, we may need to label sets of pixels. The labeling can be done using the unsupervised clustering algorithm, being K-Means [2] one of the most used techniques. The Forgy's K-Means clustering is a time consuming task and hence the operation needs to be accelerated for real time environment [3]. The software implementation of K-Means algorithm was not able to meet the timing requirement of the systems. To meet the increased demand software hardware co-simulation method has been proposed in literature [6]. Due to the laborious computations of K-Means, many hardware architectures [7]-[10] are proposed to accelerate the clustering process. The hardware specifications of these works vary because of the different target applications. Filho et al. propose a hardware / software co-design technique for K-Means clustering [4], which is used for image clustering for hyperspectral images. There are also methods using filtering algorithm and KD-trees for FPGA implementation [8]. The K-means SIP is proposed to accelerate image segmentation in SOC environments for embedded systems [10] which works for 5 dimensional feature vectors. In our previous works[11], Online K-Means clustering was developed for handling vector dimension up to 8 with maximum of 16 clusters. The online architecture developed provide the clustering result in single iteration with less memory overhead for storing the input vectors but however, the clustering quality can be improvised by Forgy's K-means

Grenze ID: 02.ICCTEST.2017.1.172

algorithm as the optimization process is dependent on the number of iterations. Hence the proposed paper addresses the issue by using the Forgy's clustering algorithm for enhancing the clustering quality and also provides effective resource utilization by adaptable hardware which adapts itself for different dimensions up to 16.

II. K-MEANS ALGORITHM-OVERVIEW

K-Means clustering algorithm regards the i-th input data xi as a D dimensional feature vector, which is represented as

$$xi = (xi,1,. . . , xi,D) \quad (1)$$

where D is the number of vector dimensions.(D<= 16).

A. Algorithm

Step 1: No. of clusters = K. Hence, K vectors are randomly selected to be centroids of the K clusters.

Step 2: Read the input vector and assign the input vector to the nearest centroid. After all the inputs are read and put into the nearest centroid, go to step 3.

Step 3: Centroid must be re-computed in each cluster using mean function as

$$\text{New Centroid} = \frac{\text{Summation of input vector in each cluster}}{\text{No. of pixel in each cluster}} \quad (2)$$

Step 4: Check whether the maximum iteration has reached. If yes, stop else go to step 2.

For multimedia analysis it is required to analyze high dimensional vectors greater than five with proper resource utilization .Owing to hardware-cost constraints in embedded systems, the proposed hardware is designed to deal with clustering problems with feature vectors whose dimension number less than or equal to 16. Depending on the feature vector size and number of clusters, the hardware resources are made adaptive so that the system resources can be utilized more efficiently. The architecture uses a Manhattan distance calculator which does not require multipliers and hence the clustering operation can be executed faster. The division module realized for finding the new centroid carries out the operation in a single clock cycle which in turn speeds up the clustering operation. Furthermore, parallel computation is used to process the data and to allocate feature vectors to different computing resources. Also the random initialization method will be implemented in the hardware modules to accelerate the processing speed and to provide appropriate solutions for clustering. algorithms proposed, we have implemented the random initialization method in the paper.

III. PROPOSED ARCHITECTURE

The proposed K-Means architecture is designed to work under the processing platform where the width of bus is 128 bits. The bit-width of each dimension of the input vector is assumed to be 8-bit, and vectors with the maximum dimension number (16 dimensions) exactly fit the width of the 128-bit system bus. One of the main contributions of the proposed architecture is the division operation which is carried out in a single clock cycle.

An overview of the proposed K-Means clustering architecture is illustrated in Figure 1, and the functionality of each module will be explained in the following subsections.

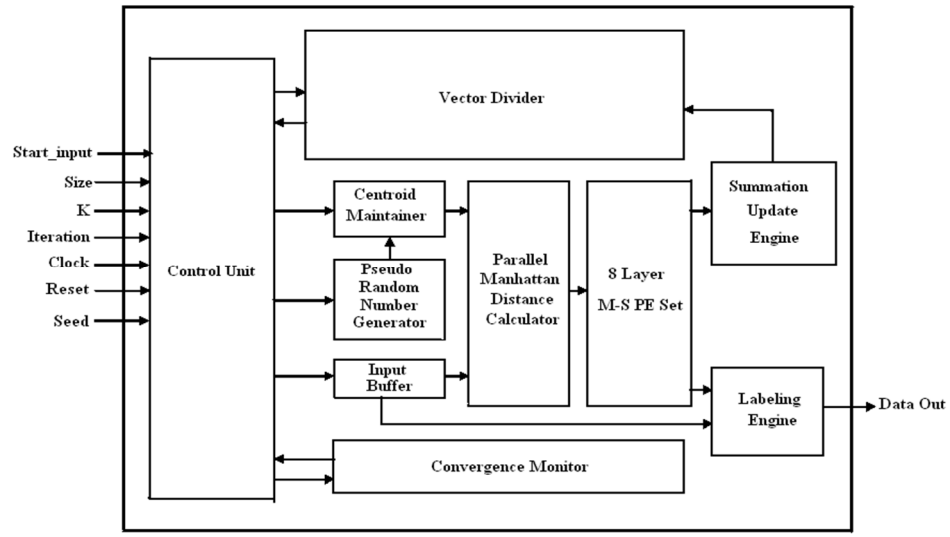

Figure 1. Proposed Architecture

A. Pseudo Random number generator

"Pseudo Random Number Generator" module generates 256 random numbers and stores them into a local buffer. Depending on the feature vector dimension D and the cluster number K, K vectors of D dimensions are selected from local buffer as initial centroid and are stored in "Centroid Maintainer" module which has to be updated by iterative steps.

The design of random sequence of numbers is done using Linear Feedback Shift Register (LFSR) as shown in Figure 2. Next number is obtained by 'hashing' the previous number with a bunch of XOR gates. In Fig 2, The LFSR register is initially loaded a zero value and bits from zero to six of this register is given to NOR gate and the output of the NOR gate is stored in a register called bits_6_zero. Next the 7^{th} bit of LFSR register and the result stored in bits_6_zero is xored and stored in feedback register. The initial number called seed is stored in taps register. If the $(n-1)^{th}$ bit of taps register is zero, then $(n-1)^{th}$ bit of LFSR_register is stored in nth bit of Next_LFSR register, else the bit from feedback register is xored with $(n-1)^{th}$ bit of LFSR_register and then the result is stored in nth bit of Next_LFSR register. The 0^{th} bit of Next_LFSR register is stored with the content of feedback register.

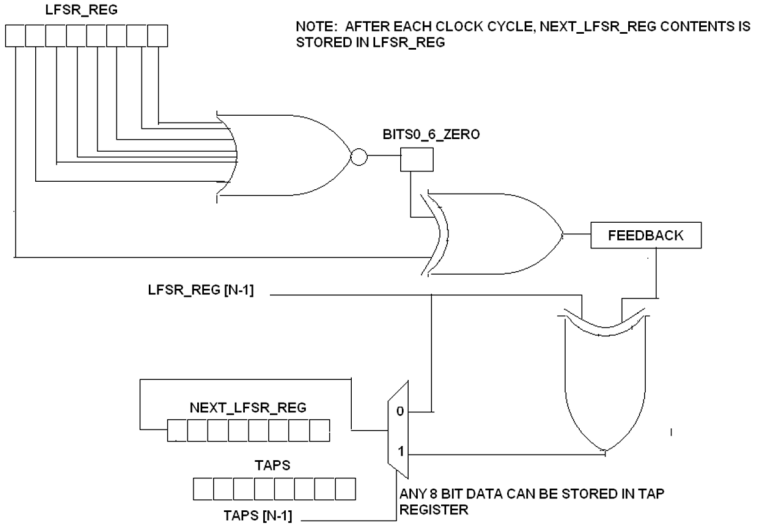

Figure 3. Design of Pseudo random number generator.

B. Pseudo Random number generator

"Parallel Manhattan Distance Calculator" contains 256 "Distance Calculator" modules, which can calculate the distance of input vectors and centroids. The main architecture of "Manhattan Distance Calculator" module is shown in Figure 3.

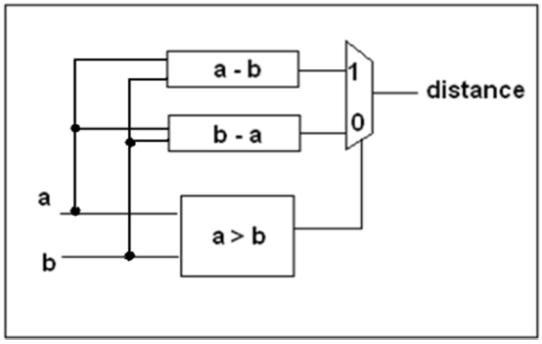

Figure 3. Manhattan distance calculator.

The module simply computes the absolute difference of the input vector and the corresponding centroid. Note that each "Distance Calculator computes the distance of only one dimension of vectors and centroids. Note that each "Distance Calculator computes the distance of only one dimension of vectors and centroids and stores it in a memory called distance memory.

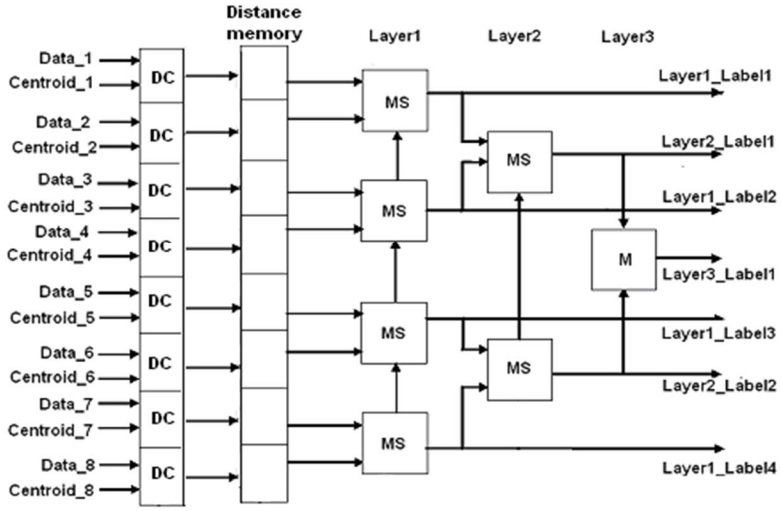

Figure 4. Architecture of 3 layer of MS-PE set

In Figure 4, "Data 1" and "Data 2" both contain the same 1-dimensional input vector. "Centroid 1" and "Centroid 2" contain the 1-dimensional vector of the first centroid and the second centroid respectively. A total of eight "Manhattan distance calculator" modules are configured as four parallel processors which can compute the distance of four 1-dimensional input vectors to their corresponding centroids simultaneously. If the modules deal with 2-dimensional vectors, "Data 1" and "Data 3" both contain the first dimension of the same 2-dimensional input vector, and "Data 2" and "Data 4" both contain the second dimension of the same 2-dimensional input vector "Centroid 1" and "Centroid 2" contain two dimensions of 2-dimensional vector of the first centroid, and "Centroid 3" and "Centroid 4" contain the two dimensions of the 2-dimensional vector of the second centroid. Eight "Manhattan distance calculator" modules are configured as two parallel processors which can compute the distance of two 2-dimensional input vectors and their corresponding centroids simultaneously. Similarly, if the module deals with 4'D vectors, then modules are configured as one processor to compute the distance of one 4-dimensional input vector and its corresponding centroid. As

we observe from the above mentioned examples that as the input vector dimension is increased the processor number decreases which indicates that the processing speed depends on input vector dimension D and cluster number K, we can observe the adaptive nature of the hardware which makes sure that all the 128 bus bits are always utilized by either increasing the number of input data or input data dimensions, from the above example. The proposed architecture is an extension of the given example, and it can deal with input vectors with dimension $D = 16$ and cluster number $K = 16$.

C. 8 layer parallel M-S PE

"Eight Layer Parallel M-S PE" module contains a set of tree structured "Minimum-Summation Processing Element modules in 8 layers. For a *L*-layer module, the number of "M-S PE" module is $2^L - 1$, and a 3-layer example is shown in Figure 4, where there are seven "M-S PE" modules in three layers. The main architecture of "M-S PE" module is shown in Figure 5. This processing element can be configured to execute two kinds of operations, "Minimum" or "Summation."

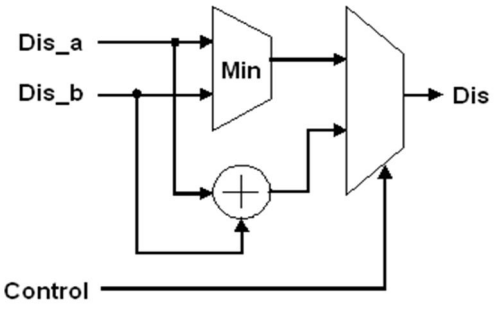

Figure 5. Architecture of Minimum-Summation element

To explain the functionality of "8-Layer Parallel M-S PE" module, the three layer M-S PE module of Figure 4 is used as an example.

For one dimension input vectors, the operation of "M-S PE" module in the first layer are set to "Minimum," and four labels, containing the nearest neighbor information of input vectors, are sent to the output signals. When the input vector dimension is two, the "M-S PE" modules in the first layer are set to "Summation," and the operations of "M-S" modules in the second layer are set to "Minimum." The distances of two pairs of 2-dimensional vectors are calculated in the first layer, and two labels, containing the nearest neighbor information of input vectors, are sent to the output signals. Similarly, when the inputs are 4'D, the distance of a pair of 4-dimensional vectors is calculated in the first and the second layer, and finally one label is sent to the output signal. Note that "M-S PE" modules in the last layer only has "Minimum" operation in this 3-layer example, and similarly, "M-S PE" modules in the last four layers only have "Minimum" operation in the proposed 8-layer architecture. In addition, in order to adjust the cluster number *K*, the redundant centroids are set to be inactive in these "M-S PE" modules, and inactive centroids are not chosen to be updated in the next stage.

D. Summation Updating Engine

The layers in the MS module contain the minimum distance between the input and the centroid. The "Summation accumulation buffer" module checks which input is near to which centroid and accumulates the corresponding input in the same centroid to compute the sum dimension wise and stores the sum in a local buffer and correspondingly increment the count. The functionality of the module is to update the summation and count of corresponding clusters, which are the numerator and the denominator in equation 2 respectively. The summation and count of each centroid are stored in a local buffer until "Vector Divider" module in the next stage requires data to perform division operation. The design of the module is as shown below in Figure 6

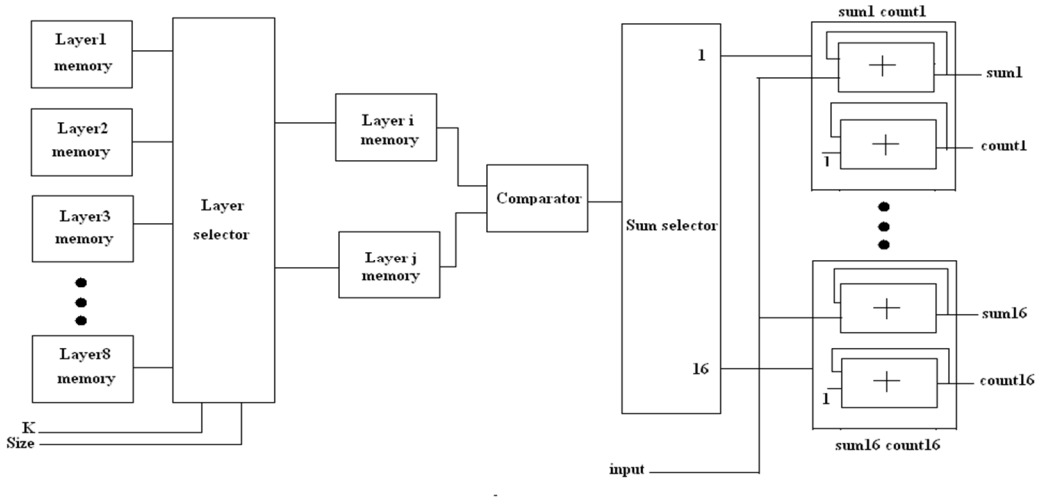

Figure 6. Architecture of Summation update engine

Depending on the input vector dimension size and the cluster number K, any of the two layers is selected for comparison of the input with the K centroids. If the input is nearer to the 2nd centroid, then the input is added with the sum2 and the count2 is incremented by one. Similarly, all the inputs are checked and it is summed in the corresponding centroid

E. Vector divider

"Vector Divider" module is able to compute a division of a 16 dimensional vector. The division operation is performed in a single clock cycle irrespective of the number of inputs in each cluster. The new centroids calculated are fed to "Centroid Maintainer" module for the next iteration.

The division operation is performed as multiplication of (1/count) * sum. 1/count is realized as follows

- Set the precision Pr to be initially 1.
- Check the value of N. If N<10, then set the Numerator Nr =10, else if N<100 set the Numerator Nr=100. Similarly check for N less than 1000, 10000, 100000 and 1000000 and correspondingly set the Numerator value Nr to be 1000, 10000, 100000 and 1000000 respectively.
- Set current iteration value i to be 1.
- Perform N*i
- Check if N*i is less than Nr. If true then increment i and go to step 4.
- Set the quotient Q_{pr} to be i-1 and remainder R to be Nr mod N.
- Check if precision Pr is less than or equal to 3. If true then increment Pr and go to step 3.
- Output the four bit precision quotient mantissa $Q_1Q_2Q_3Q_4$ and exponent 10^{-4}.

The four bit quotient obtained is in the BCD form. Hence we need to convert the BCD quotient to binary for further processing. The process of conversion again involves multipliers and adders. The binary equivalent of the quotient is obtained by computing $Q_{fianl} = (Q_1 * 1000) + (Q_2 * 100) + (Q_3 * 10) + Q_4$. The obtained quotient Q_{fianl} is then used to compute new centroid.

F. Convergence Monitor and Labeling Engine

"Convergence Monitor" module examines whether the iteration is finished and returns the information to the "Control Unit" module. The functionality of "Labeling Engine" module is to output the label or the vector of the nearest centroid of each input vector.

IV. RESULTS

The result contains three parts. The first shows the simulation results for the K-Means hardware depicted Figure 7, the second is the snapshots of synthesized netlist using Cadence RTL compiler shown in Figure 8. The third is the comparison table illustrating the hardware specifications.

The first part shows adaptive nature of the proposed architecture which is shown in Figure 7a and Figure 7b. Figure 7a shows the simulation results for input vector dimension D=2 and Cluster number K=2 for this the hardware is configured to handle 64 input data at a time. Figure 7b shows the simulation results for input vector dimension D=5 and Cluster number K=4 for this the hardware is configured to handle 8 input data at a time. Also it is seen in simulation result that the division operation is performed in single clock cycle which enhances the speed of clustering. The synthesized netlist for MS-PE block is shown from Figure 8.

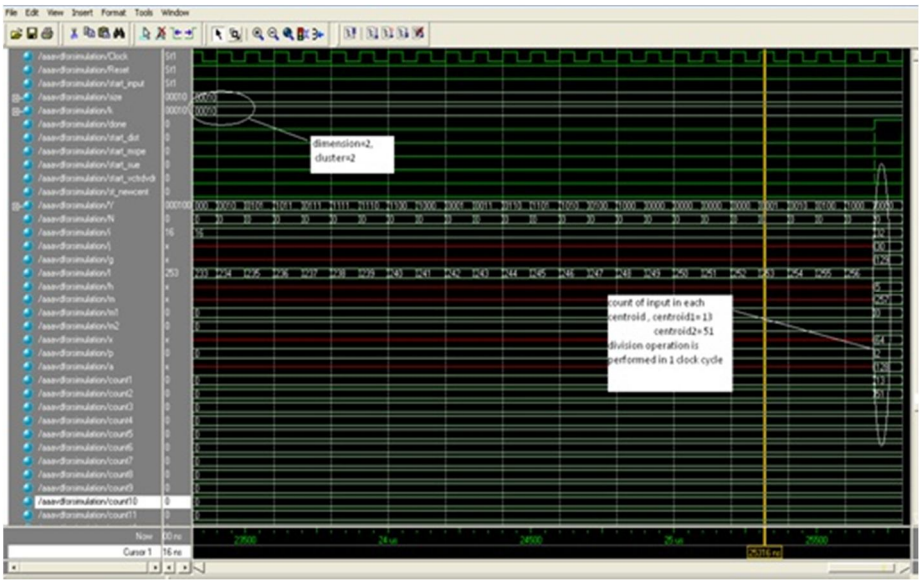

Figure 7a Simulation Results with D=2 and Cluster number K=2

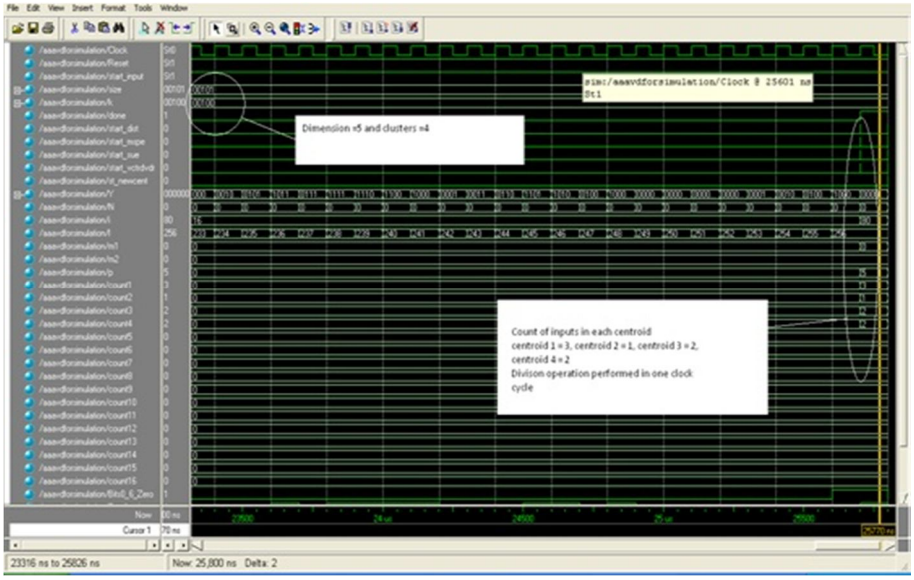

Figure 7b Simulation Results with D=5 and Cluster number K=4

The third part is the comparison of hardware specifications. This work is compared with work of [13]. The proposed architecture performs the division operation in single clock cycle and hence takes lesser time to converge and hence is faster. Also, the proposed work has resource adaptive mechanism to handle vectors with different dimensions efficiently. Since the proposed architecture uses Manhattan distance calculator, the space occupied is less as it does not require multiplier.

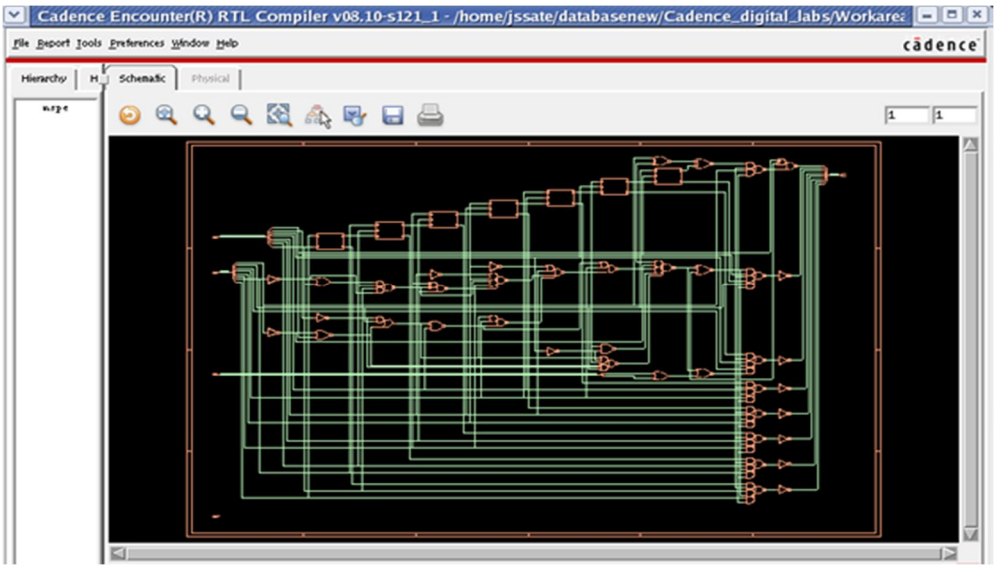

Figure 8. Synthesized netlist of MS_PE block

TABLE I. NOTE HOW TABLE IS LEFT ALIGNED

Specifications	TVLSI 2010	This Work
Technology	TSMC 180nm	GPDK 180nm
Distance measurement	EqulidianManhattan	Manhattan
Division operation performed in	Less than10 cycles	1 cycle
Maximum Throughput	16 dimensions/cycle	16 dimension/cycle

V. CONCLUSIONS

K-Means is an important clustering algorithm in the field of pattern recognition and data mining. To make this algorithm feasible for multimedia applications like image analysis in real-time embedded systems, a resource adaptive scheme and random initialization of centroids with single cycle clock divider for K-Means architecture is proposed in this paper. The architecture is designed for different dimension numbers feature vectors by configuring the processing elements.

REFERENCES

[1] W. Al-Khatib, Y.F. Day, A. Ghafoor, and P.B. Berra. Semantic modeling and knowledge representation in multimedia databases. IEEE Transactions on Knowledge and Data Engineering, 11(1):64–80, Jan/Feb 1999

[2] J. B. MacQueen,1967 “Some methods for classification and analysis of multivariate observation”, In: Le Cam, L.M., Neyman, J. (Eds.), University of California.

[3] T.-W. Chen, Y.-L. Chen and S.-Y. Chien, “Fast image segmentation based on K-Means clustering with histograms in HSV color space,” in Proceedings of IEEE International Workshop on Multimedia Signal Processing, Oct. 2008, pp. 322–325.

[4] A.G. d. S. Filho, A. C. Frery, C. C. de Araújo, H. Alice, J. Cerqueira,J. A. Loureiro, M. E. de Lima, M. d. G. S. Oliveira, and M. M. Horta, “Hyperspectral images clustering on reconfigurable hardware using the K-means algorithm,” in *Proc. Symp. Integr. Circuits Syst. Des.*, Sep.2003, pp. 99–104.

[5] Abbo, R. Kleihorst, V. Choudhary, L. Sevat, P. Wielage, S. Mouy,and M. Heijligers, “XETAL-II: A 107 GOPS, 600 mW massively-parallel processor for video scene analysis,” in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, Feb. 2007, pp. 270–271.

[6] AC Frery, CCde Araujo,H Alice “Hyperspectral images clustering on reconfigurable hardware using the K-Means algorithm” in *Proc. IEEE Int. Symp. Circuits Syst.*, Sep 2003, pp. 94–104

[7] Maliatski and O. Yadid-Pecht, “Hardware-driven adaptive K-means clustering for real-time video imaging,” *IEEE Trans. Circuits Syst.Video Technol.*, vol. 15, no. 1, pp. 164–166, Jan. 2005.

[8] Huang and D.-H. Liu, “Segmentation of color image using EM algorithm in HSV color space,” in *Proceedings of IEEE International Conference on Information Acquisition*, Jul. 2007, pp. 316–319.