Coordinate Rotation Based Low Complexity K-Means Clustering Architecture

Bhagyaraja Adapa, Dwaipayan Biswas, Swati Bhardwaj, Shashank R, Amit Acharyya, *Member, IEEE*, Koushik Maharatna, *Member, IEEE*

***Abstract*—In this paper, we propose a low-complexity architectural implementation of the K-Means based clustering algorithm used widely in mobile health monitoring applications for unsupervised and supervised learning. The iterative nature of the algorithm, computing the distance of each data point from a respective centroid for a successful cluster formation until convergence presents a significant challenge to map it onto a low-power architecture. This has been addressed by the use of a 2-D Coordinate Rotation Digital Computer (CORDIC) based low-complexity engine for computing the n-dimensional Euclidean distance involved during clustering. The proposed clustering engine was synthesized using the TSMC 130 nm technology library and a place and route was performed following which the core area and power were estimated as 0.36mm^2 and 9.21mW @ 100 Mhz respectively making the design applicable for low-power real-time operations within a sensor node.**

***Index Terms*—K-Means, CORDIC, signal processing, hardware design, low complex architecture.**

I. Introduction

THE fundamental concept of cluster analysis is to form groups of similar objects as a means of distinguishing them from each other [1]. Clustering techniques have been successfully used in diverse fields such as medicine (EEG, Functional MRI, activity recognition), geography or marketing, involving multivariate data and can be conveniently deployed with limited resources (memory and CPU) [1]. The K-Means clustering algorithm owing to its computational simplicity, efficiency has been an attractive choice for a wide variety of signal processing applications [1]. It is a well-perceived fact in the research community that cluster analysis is primarily used for unsupervised learning where the class labels for the training data are not available. However, the K-Means algorithm can also be used for supervised learning where the class labels of the training data are known a priori [3]-[9]. Apart from using it as a learning algorithm, K-Means has also been utilized for signal pre-processing, feature reduction and time-domain signal analysis [2] . Hence, using K-Means for real- time cluster analysis requiring computation in resource constrained sensor nodes for remote health care monitoring systems where online multi-modal data acquisition and analysis is the key (e.g. cardiovascular disease prognosis), requires an effective implementation strategy. The fundamental requirement for such applications is to cut-down on continuous transmission and have a low-power operation to prolong their battery life. Hence, in view of this an effective algorithm-to-architecture holistic mapping is required to fulfill the notion of low-power operation aimed for long durations.

The K-Means algorithm exhibits an iterative nature, where it computes the distance of each data sample from the centroids until convergence. This is generally achieved by the use of power hungry multipliers, square rooters (for Euclidean distance computation) and multiplexers [3]-[9], thereby rendering direct mapping of this algorithm to architecture infeasible for implementation on resource-constrained platforms. An attempt was made to replace the Euclidean distance by a combination of Manhattan and Max distance but by trading-off accuracy for power consumption in [10]. Therefore, an algorithm-to-architecture holistic optimization approach is necessary for maintaining its algorithmic efficiency and making it low-complexity from the architectural perspective. Coordinate Rotation Digital Computer (CORDIC) based architectures exploring its different transcendental functions to compute complex arithmetic operations [11]-[12] have been used widely for computationally intensive signal processing algorithms [11]-[14]. K-Means clustering algorithm has been pre-dominantly used with fore-mentioned algorithms [11]-[14]. Hence, in this study, we investigate the use of a CORDIC-based low-complexity engine to implement K-Means clustering algorithm.

The rest of the paper is organized as follows. Section II provides with the necessary theoretical background, section III discusses about the proposed methodology, Section IV presents the hardware complexity analysis, and section V concludes the discussion.

II. Theoretical Background

With a given dataset X = x_i, i = 1,...,n to be clustered into a set of k clusters, the K-Means algorithm iterates to minimize the squared error between the empirical mean of a cluster and the individual data points, defined as the cost function,

$$J(\theta, u) = \sum_{i=1}^{n} \sum_{j=1}^{k} u_{ij} (x_i - \theta_j)^2 \tag{1}$$

where, θ_j is the cluster center and u_{ij} = 1 if x_i lies close to θ_j, or 0 if otherwise. Initially k centroids are defined and data vectors are assigned to a cluster label depending on how close they are to each centroid. The k centroids are recalculated from the newly defined clusters and the process of reassignment of each data vector to each new centroid is repeated. The algorithm iterates over this loop until the data vectors from dataset X form clusters and cost function J is minimized [2].

B. A, S. B, S. R and A. A are with the Department of Electrical Engineering, Indian Institute of Technology Hyderabad, 502285, India, e-mails: {ee13m1019, ee14resch11018, ee12b1026, amit_acharyya}@iith.ac.in. D. B and K. M are with School of Electronics and Computer Science, University of Southampton, UK-SO17 1BJ, {db9g10, km3}@ecs.soton.ac.uk.

The project is partly funded under the Early Carrier Research Grant from the Science and Engineering Research Board, Govt.of India with Sanction No: ECR/2015/000148 dated 14th July, 2016. S. B's PhD fellowship was funded under Internet Of Things (IOT) for smarter Healthcare project under Grant No: 13(7)/2012-CC&BT, MEITY, Govt. of India dated 25th February, 2013. All VLSI CAD tools used in this work were supported under SMDP-C2SD, Ministry of Electronics and Information Technology, Govt. of India.

CORDIC is an efficient implementation technique for vector rotation and arctangent computation. Since it can be realized using simple shift and add operations, it is very effective in terms of low hardware complexity [11]. Considering the rotation in the clockwise direction, the basic CORDIC expressions can be expressed as

$$\begin{bmatrix} x_f \\ y_f \end{bmatrix} = \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \quad (2)$$

where x_0, y_0 and x_f, y_f are the initial and final components of the vector and the angle of rotation is θ. In this paper, we use CORDIC in vectoring mode for our implementation.

III. Proposed Methodology

The architecture of the proposed CORDIC based K-Means clustering engine is shown in Fig.1. The input data is stored in the memory unit for further usage and are transmitted to different blocks via control unit (CU). Memory unit also serves the purpose of storing intermediate values. The Euclidean distance is calculated in distance unit (DU) using low complexity CORDIC vectoring module. The distances from each point to each of the centroids are sent to a comparator block to identify the cluster to which it belongs. Once the clustering is done, centroid calculation block will be activated to compute the new centroids. If these new centroids differ significantly from the previous iteration values, then clustering will be repeated, else clustered data will be sent to the output. The CU governs the data flow among all the modules. The proposed engine utilizes CORDIC to compute Euclidean distance between two points, which is a metric to compute the clusters and has been explained through an illustrative example.

In this study, our focus is to propose a methodology for utilizing CORDIC to compute Euclidean distance between two points, which is a metric to compute the clusters. In two dimensional signal space, if (x_1, x_2) , (y_1, y_2) are two points, the Eucliean distance between these two points will be

$$dist_{2D} = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

One square rooter, two square, one adder and two subtraction operations are involved in this computation. If we give a and b as the x- and y-inputs to the vectoring mode CORDIC, the x output will be the magnitude of Vector(a, b), which is $\sqrt{a^2 + b^2}$. So, with $(x_1 - y_1)$ and $(x_2 - y_2)$ as the x- and y- inputs repectively, the vectoring mode will give distance between the two points. Architecture of the 2D distance measurement unit using vectoring mode CORDIC is shown in the Fig. 2(a). We can extend this methodology to n-dimensional(nD) signal space to formulate distance between two nD vectors. Considering the case of 3-D signal space (n=3), distance between these two points $(x_1, x_2, x_3), (y_1, y_2, y_3)$ will be

$$distance_{3D} = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2}.$$

With inputs as $(x_1 - y_1)$ and $(x_2 - y_2)$ to vectoring mode as in 2-dimensional case, the x-output of vectoring mode CORDIC(vec_x^{l1}) is represented as :

$$distance_{2D} = vec_x^{l1}((x_1 - y_1), (x_2 - y_2)).$$

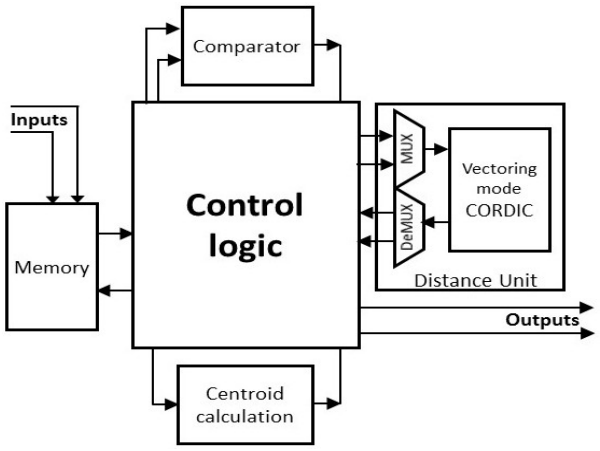

Fig. 1. Complete architecture of the proposed system.

If we pass this x-output to next CORDIC level as one input, with $(x_3 - y_3)$ as second input, x-output will be the desired 3-dimesional distance

$$distance_{3D} = (vec_x^{l2}(vec_x^{l1}((x_1 - y_1), (x_2 - y_2)), (x_3 - y_3)).$$

Fig. 2(b) shows the 3D distance measurement unit. Since these two stages are executed sequentially, same CORDIC unit can be reused only at the expense of two multiplexers as shown in Fig.2(c). For two n-dimensional vectors $(x_1, x_2, x_3, \ldots, x_n)$ and $(y_1, y_2, y_3, \ldots, y_n)$, the $distance_{nD}$ between them,

$$distance_{nD} = \sqrt{(x_1 - y_1)^2 + \cdots + (x_n - y_n)^2}$$

can be calculated using (n-1) CORDIC stages as shown in Fig. 2(d). For calculating distance between two n-dimensional vectors, the inputs to the vectoring mode CORDIC block will be as follows. Thus, recursively using the fundamental 2D-CORDIC unit we can generalize it for computing the n-dimensional distance. As the calculations are done sequentially, the same CORDIC unit can be reused for calculations in two levels, only at the expense of one 2-input multiplexer and one (n-1) input muliplexer at inputs of CORDIC unit (as shown in fig. 2(d)). Multiplexers at input and Demultiplexers at output of the CORDIC block have been used accordingly.

For 1st level of CORDIC (vec_x^{l1}) x-input: $(x_1 - y_1)$, y-input: $(x_2 - y_2)$

For pth level of CORDIC (vec_x^{lp}) x-input: previous level x-output, y-input:$(x_p - y_p)$,where $p = 2$ to n

IV. Results, Analysis and Discussion

A. Methodology Validation

The proposed architecture was coded in Verilog as HDL and functionally verified by evaluating on a set of 24 datasets each having 60 samples of kinematic data, collected from a wrist-worn tri-axial accelerometer measuring human arm movements. This dataset was chosen in view of the popularity of K-means for analyzing human movement in daily living scenarios using inertial sensors [15]. The proposed methodology can be implemented with different specifications. Here, we have verified the design for cluster values ranging from 1 to 16 (K) and dimension of the input data as 3 (n). It is

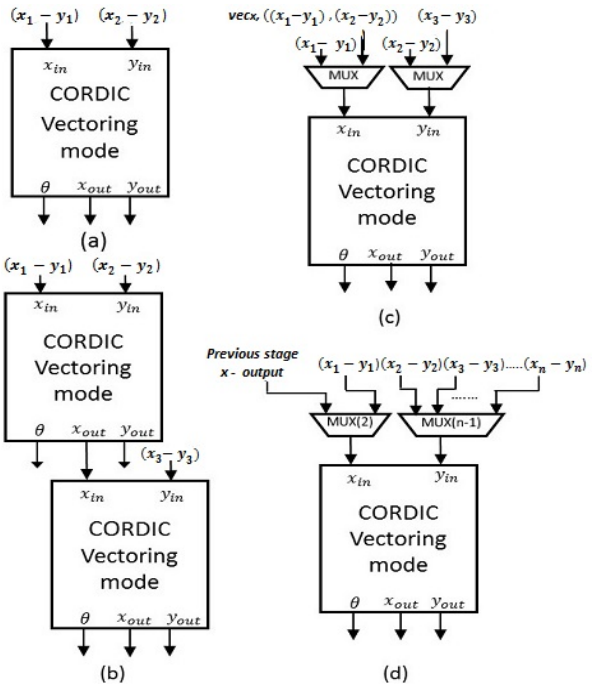

Fig. 2. CORDIC based distance measurement (a)2-D vectoing;(b)3-D vectoring;(c)3-D multiplexed architecture;(d)n-D multiplexed architecture.

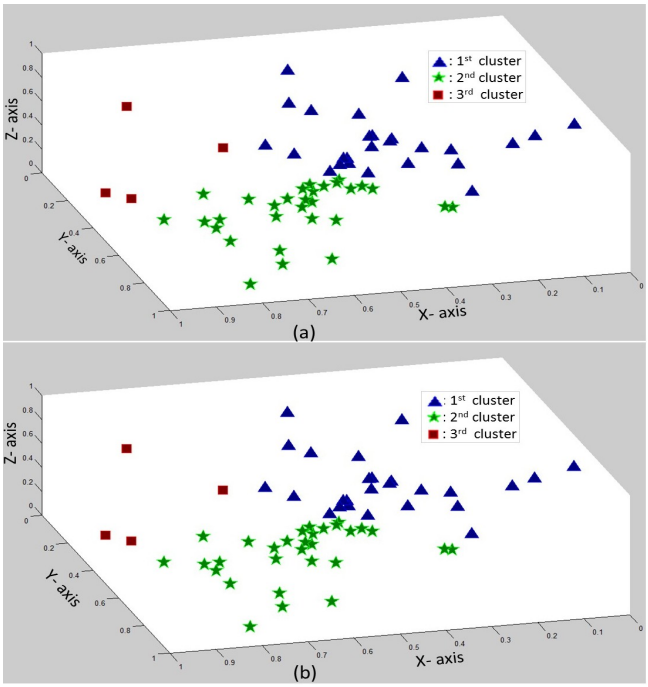

Fig. 3. 3D plot of the data points clustered using (a) inbuilt kmeans function; (b) CORDIC based kmeans function.

important to note here that although the input data is 16-bits wide, the width of the datapath in the CORDIC unit is 22-bits. In order to achieve the desired 16-bit accuracy a 22-bit word-length should be selected, according to the formulation (N + Log2N + 2) and having atleast 16 iterations. Therefore, to obtain a high accuracy a 22-bit CORDIC was used for this implementation. The output was validated against Matlab model of the K-Means algorithm with same initial seed values for both the cases. The results indicate the similarity in predicting the number of clusters and the iterations taken by both the approaches. Moreover, we achieve precision up to 8 decimal places using a 16-stage vectoring mode CORDIC with input magnitudes ranging from 1 to 10^{15}. Fig. 3(a) shows the resulting clustered data from Matlab inbuilt k-means function and Fig. 3(b) shows the clustered data output using the proposed methodology. The comparison of Fig. 3(a) and Fig 3(b) shows that the proposed methodology produces 100% accuracy, exhibiting a robust system. A detailed error analysis has been provided in Section C.

B. Hardware Complexity Analysis

Throughout the hardware complexity analysis, we keep a generalized view of word-length b and followed the same procedure used in [11]. Since the distance computation in K-Means Clustering using CORDIC is an iterative procedure, we consider only one single iteration because the same hardware can be reused for the next iterations as well as for successive stages of CORDIC in Vectoring Mode for dimensions higher than 2.

Computation of distance between 2 n-dimensional points $(x_1, x_2, x_3, \ldots, x_n)$ and $(y_1, y_2, y_3, \ldots, y_n)$ using the conventional method, i.e. $\sqrt{(x_1-y_1)^2 + \cdots + (x_n-y_n)^2}$ requires n squaring operations,$(n-1)$ addition operations and 1 square-root operation. Since this distance is only used for comparison, the absolute value of the distance is not required and hence, square rooting operation can be omitted without compromising on accuracy. To provide a comparison on a uniform platform, we consider only Ripple Carry Adder (RCA) and Conventional Array Multiplier (CAM) as the means of implementing the arithmetic operations. One $b-$bit RCA requires b full adders (FA) (in a simplified view) [11], and b X b CAM requires $b*(b-2)$ FA plus b half adders (HA) and b^2 AND gates [11]. Similarly, one b-bit SQRT needs 0.125 $*$ $(b+6)b$ FA and XOR gates [11]. In addition, considering one FA cell requires 24 transistors, one HA cell, and one two input XOR gate consist of 12 transistors and a two input AND gate consists of six transistors [11], we can calculate TCA = $24b$, TCM = $6b(5b-6)$, where TC* are the transistor counts for RCA and CAM respectively. Following the same procedure used in [11], savings in terms of arithmetic operations for distance computation in different dimensions without using CORDIC are computed. For n-dimensional distance computation, $(n-1)$ RCAs and n CAMs are required. The transistors used here will not be required when the proposed CORDIC based engine is used for K-Means clustering, since we are reusing the CORDIC unit. Therefore, the total Transistor Count (TC) computed here will be the Transistor Saving (TS), given by:

$$TS_{nD} = nTC_M + (n-1)TC_A$$

Expressing TS_{nD} in terms of total number of transistors saved and normalizing with respect to b, a metric Transistor Saving per Word-length (TSPW) can be computed following the approach presented in [16]. Being the function of b and n, the Figs. 4 and 5 show that the increase in TSPW for the proposed CORDIC based K-Means Clustering methodology is significantly higher than the conventional design for two

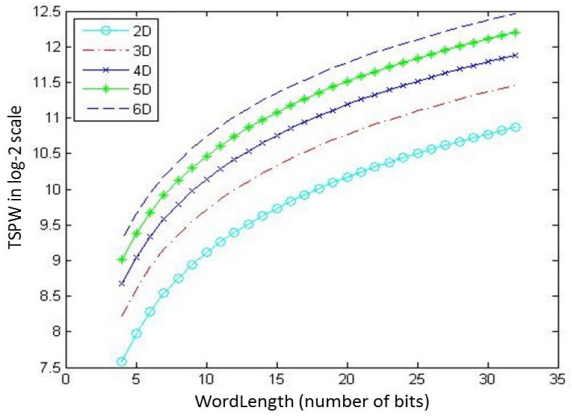

Fig. 4. Variation of Transistor saving per word-length of the proposed algorithm with word-length and dimension.

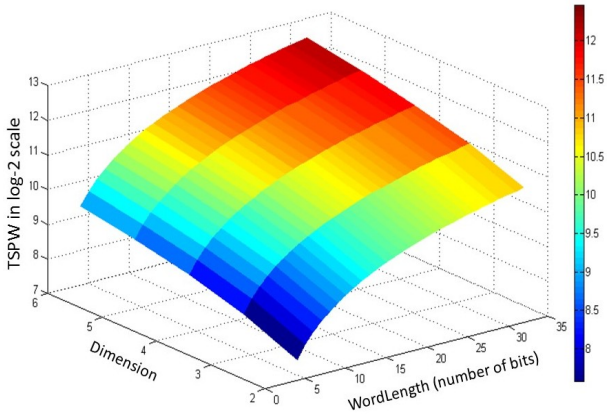

Fig. 5. Comparative variation of Transistor Saving Per Word-length (TSPW) of the proposed algorithm word-length for 2D to 6D.

dimensional to six dimensional $((184.5 * b - 69)TSPW)$ points, for different word lengths $(4 <= b <= 32)$.

C. Error Analysis

Accuracy is determined by comparing outputs from proposed methodology with Matlab inbuilt 'kmeans' function. In the proposed technique CORDIC is used for calculating the Euclidean distance. The accuracy depends on the input dimension (D) and the number of micro-rotations, which is equal to number of stages(n) of CORDIC. A set of 1000 randomly generated signals are taken as input and Mean Absolute Percentage Error(MAPE) was calculated with various stages of CORDIC for different input dimensions. Fig.6 shows the variation of MAPE for the proposed architecture with dimensions ranging from 2D to 6D, with different CORDIC stages n = 8, 12, 16 and 20, and with word length b = 4, 8, 16 and 32. MAPE is of the order of 10^{-2} with number of CORDIC stages(n) as 8 for a given dimension while with n=16, MAPE decreases to the order of 10^{-4}. From Fig.6 it is evident that as number of CORDIC stages increases, MAPE decreases due to the increase in resolution of CORDIC. Furthermore, for a fixed number of stages of CORDIC, MAPE increases with the dimensionality of the input because subsequent dimensions are passed on to CORDIC in a cascaded fashion.In addition, from Fig.6 it is observed that for a given dimension and number of CORDIC stages, MAPE does not change significantly with the word length b. It is important to

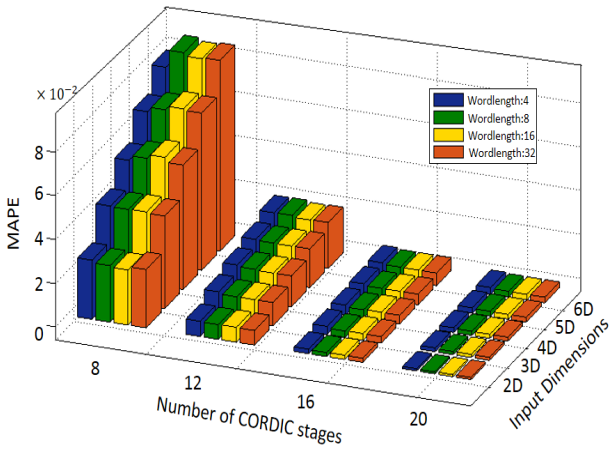

Fig. 6. Variation of Average Percentage error of the proposed algorithm with number of CORDIC stages, dimension and wordlength .

note that although Fig.6 has been demonstrated for processing data upto 6D, the architecture can support upto 16D input data. For a 16 stage CORDIC, all the stages are sequential and at any instant of time only one of these 16 stages work on the particular data point. Hence, 16 different data points can be processed using 16 stages of CORDIC at any clock cycle. Moreover, this (MAPE) is the error in Euclidean distance which is used to cluster data points. The final output is the co-ordinates of the cluster centroid and not the Euclidean distance itself, which ensures a robust output.

D. Analysis of computational time

In the proposed architecture, an n stage (=16) CORDIC takes n number of clock cycles to compute a single two dimensional Euclidean distance and therefore , achieves 100% throughput with 16 dimensions/cycle(cf. Table 1), having a latency of n clock cycles (=16), equivalent to the number of stages of CORDIC operation.

For each cluster (K), distances from every cluster centroid to every data point have to be measured. To calculate n dimensional distance, $(n-1)$ stages of CORDIC are needed. Hence, each iteration takes $k*num*(D-1)+16$ clock cycles, where num is number of data points, D is dimensionality of data and k is number of clusters. The variation of required number of clock cycles with varying dimensions and number of clusters for a dataset having 64 points is shown in Fig.7. For clustering 64 datapoints into 3 clusters, a 16 stage CORDIC distance measuring unit takes 400 clock cycles. The proposed architecture at $100MHz$ operating frequency will take $4us$ to complete a single iteration step. Here, the number of iterations depend on the initial seeds and the distribution of data. Considering a worst case scenario of 1000 iterations, the total K-Means operation using the proposed architecture takes less than $1ms$ to complete, thereby achieving real-time standards. In Fig.8, we present a relationship for a range of functionally verified input frequencies (1 MHz to 360 MHz) against the surface power density (SPD - power per unit area) obtained as a result of DC synthesis using TSMC 130 nm technology library. A third order polynomial fit is used to describe this relationship, over the selected frequency range wherein we achieve a 100% throughput using

TABLE I
COMPARISONS OF THE PROPOSED ARCHITECTURE WITH OTHER REPORTED ARCHITECTURE

	ISCAS 08[4]	TVLSI 10[5]	TVLSI 11[6]	TMC 11[7]	ETCAS 11[8]	TCAS 13[9]	Proposed
Technology	TSMC 180nm	TSMC 90nm	TSMC 90nm	TSMC 90nm	TSMC 90nm	TSMC 90nm	TSMC 130nm
Clock Frequency	200MHz	400MHz	333MHz	500MHz	625MHz	400MHz	100MHz
Area	$0.58mm^2$	$1.23mm^2$	$1.16mm^2$	$0.67mm^2$	$0.14mm^2$	-	$0.36mm^2$
Normalized Area*	$0.15mm^2$	$1.23mm^2$	$1.16mm^2$	$0.67mm^2$	$0.14mm^2$	-	$0.1725mm^2$
Number of K	$1-16$	$1-16$	1024	$1-64$	$1-4$	$1-16$	$1-16$
Vector Dimension	$1-5$	$1-16^2$	8	$1-5$	$1-4$	$1-16$	$1-16$
Max. Throughput	2.5 D/cycle	16 D/cycle	8 D/cycle	5 D/cycle	4 D/cycle	1 D/cycle	16 D/cycle
Power	-	-	-	209mW	12.1mW	6.02mW	9.21mW
Normalized Power*	-	-	-	209mW	12.1mW	6.02mW	4.6mW
Maximum Data	2^{17}	2^{20}	2^{24}	2^{20}	2^{20}	2^{16}	2^{22}

* Normalized to 90nm from 130nm, 180nm, and the normalization factor is $(90/180)^2$, $(90/130)^2$ for 90nm and 130nm respectively, D - dimension.

the shared CORDIC resource. The reported cost function ($SPD = 8.9e^{-13}F^3 - 5.2e^{-10}F^2 + 1.8e^{-7}F + 1.2e^{-5}$) and the associated goodness-of-fit parameters namely, low values of sum of squares due to error (SSE), root mean square error ($rmse$) and high value of the adjusted R-square as mentioned in Fig.8, indicate a best fit in comparison to other tested models.

E. Comparison with other Architectures

The proposed architecture is synthesized by Synopsys Design Compiler (DC) and the place and route was performed using Synopsys IC Compiler (ICC) using $0.13\mu m$ standard cell CMOS technology. The core area and power consumption of the proposed engine are $0.36mm^2$ and $9.21mW$ at 100-MHz frequency for VDD = 1.2 V. The engine consumes 62% less power with a comparable area consumption w.r.t. state-of-the art architecture for ASIC implementation in [8] (the power reported is from back-end simulation using SoC Encounter). A comparison of the area requirement and power consumption of the proposed engine with state-of-the-art architectures have been highlighted in Table I. Since different architectures use different technologies, it is unfair to compare them on a one-to-one basis. So, area and power values are normalized to same technology node [17]. These results are provided to give an insight about the performance of the proposed-methodology-based architecture. The proposed CORDIC based clustering engine compares favorably in terms of area w.r.t. the other reported architectures as illustrated in Table I. It is to be noted that due to the unavailability of an appropriate memory module in our standard cell library, the architecture is implemented using registers. We believe that the use of appropriate memory will significantly reduce the area and power consumption.

V. Conclusion

In this paper, we proposed a novel clustering engine using CORDIC to implement the k-means algorithm. It has been generalized to compute n-D distance using a 2-D fundamental core exploiting the recursive nature of the algorithm. The hardware analysis shows a minimal amount of transistor overhead and the low power and area achieved at a relatively high clock speed (i.e. 100 MHz) makes it suitable for on-board sensor processing in pervasive healthcare applications. This engine can be suitably integrated onto a sensor platform in the form of a dedicated ASIC as first step towards point-of-care diagnostic for applications involving activities of daily living using inertial sensors where clustering techniques are an integral part of unsupervised learning approaches on datasets with no corresponding class information.

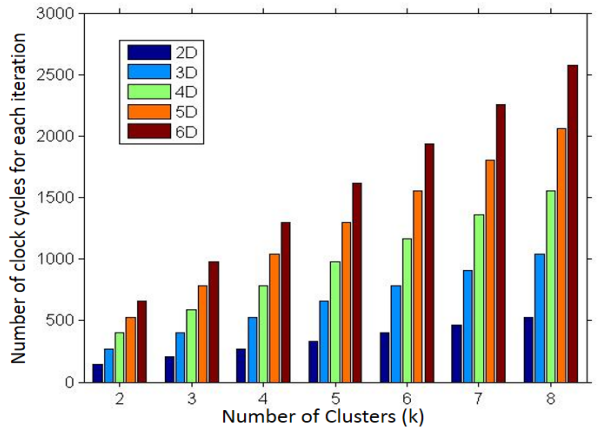

Fig. 7. Variation of required Number of clock cycles for each iteration with number of clusters and dimension of input.

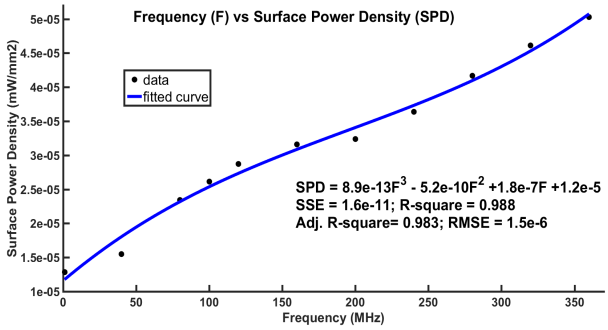

Fig. 8. Variation of Surface Power Density(SPD - synthesized power per unit area) with frequency.

References

[1] "Altera, DE2-115 User Manual," 2012.[Online]. Available: ftp://ftp.altera.com/up/pub/Altera Material/12.1/Boards/DE2-115/DE2 115 User Manual.pdf [Accessed: 01-06-2014].

[2] S. Theodoridis and K. Koutroumbas, "Pattern Recognition," 4th ed., Elsevier, pp. 30-31 and pp. 280-288, 2008.

[3] Hussain, et al.,"FPGA implementation of K-Means algorithm for bioinformatics application: An accelerated approach to clustering Microarray data," in *NASA/ESA Conference on Adaptive Hardware and Systems* , June 2011.

[4] Chen, et al., "Architectural analyses of K-means silicon intellectual property for image segmentation," in *IEEE ISCAS*., May 2008.

[5] Chen, et al., "Bandwidth adaptive hardware architecture of K-means clustering for video analysis," *IEEE TVLSI*, vol. 18, no. 6, Jun. 2010.

[6] Chen, et al., "Flexible hardware architecture of hierarchical K-means clustering for large cluster number," *IEEE TVLSI*, vol. 19, no. 8, Aug.2011.

[7] Chen, et al., "Photo retrieval based on spatial layout with hardware acceleration for mobile devices," *IEEE TMC*, vol. 10, no. 11, Nov. 2011.

[8] Chen, et al., "Power-efficient hardware architecture of K-Means clustering with Bayesian-information-criterion processor for multimedia processing applications," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* , vol. 1, no. 3, Sep. 2011.

[9] Chen, et al., "Design and Implementation of Low-Power Hardware Architecture With Single-Cycle Divider for On-Line Clustering Algorithm," *IEEE TCAS I*: Regular Papers, vol. 60, no. 8, Aug. 2013.

[10] Estlick, et al., "Algorithmic transformations in the implementation of K-Means clustering on reconfigurable hardware," *Proceedings of the 2001 ACM/SIGDA ninth international symposium on FPGA,* ACM, 2001

[11] Acharyya, et al., "Coordinate Rotation Based Low Complexity N-D FastICA Algorithm and Architecture," *IEEE Transactions On Signal Processing*, Vol. 59, No. 8, August 2011.

[12] Biswas, et al., "A CORDIC-based Low-power Statistical Feature Computation Engine for WSN Applications", *Circuits, Systems, and Signal Processing* vol. 34, no.12 (2015).

[13] Bhardwaj, et al., "Online and automated reliable system design to remove blink and muscle artefact in EEG", *IEEE, EMBC 2015*.

[14] Vemishetty, et al., "Affordable low complexity heart/brain monitoring methodology for remote health care", *IEEE, EMBC 2015*.

[15] Biswas, et al. "Recognizing upper limb movements with wrist worn inertial sensors using k-means clustering classification." *Human movement science,Elsevier*, vol. 40 2015.

[16] Acharyya, et al., "Memory reduction methodology for distributed arithmetic based DWT/IDWT0 exploiting data symmetry," *IEEE TCAS II*, Exp. Briefs, vol. 56, no. 4, Apr. 2009.

[17] Weste, et al., "Principles of CMOS VLSI design", Vol. 188. New York: Addison-Wesley, 1985.