CHARACTERIZATION OF TINY INTERPOLATED OBJECTS USING B-SPLINE COMPUTATION

[1]N Angayarkanni[*], D Kumar, [2]S Jerald Jebakumar and [3]A Albert Raj

[1]Department of Electronics and Communication Engineering, School of Architecture Engineering and Technology, Periyar Maniammai University Thanjavur, Tamilnadu, India.
[2]Department of Electronics and Communication Engineering
The Rajaas Engineering College, Vadakkangulam- 627117, Thirunelveli, Tamilnadu, India.
[3]DMI Engineering College, Aralvaimozhi - 629301, Tamilnadu, India.

ABSTRACT: In this paper a novel medical image processing system is discussed. A modified algorithm for the reduced hardware has been designed. Finally some suitable modification in the hardware is made. In Medical Image Processing System, characterization and recognization of suspended minute particles in cellular fluid is an important problem. The two key steps involved are up-sampling of the region of interest (RoI) and edge detection. In our case, the target RoI's are already blurred due to the presence of surrounding cellular fluids. So up sampling of these RoIs makes them more blurred and as a consequence, the subsequent edge detection becomes a futile exercise. Henceforth arises the requirement of an interpolation algorithm, which preserves some high pass contents. The B-Spline interpolation satisfies the above-mentioned constraint. In past, few papers dealt with the implementation of B-Spline hardware. However they failed to exploit some important properties of basis functions like periodicity. Here we have efficiently utilized those properties to achieve a novel parallel architecture.

Keywords: B-Spline, VLSI, parallel architecture and region of interest.

1 INTRODUCTION

This Paper describe about a novel medical image processing system. In past, few papers that dealt with the implementation of B-spline hardware failed to exploit some important properties of basis functions like periodicity. Here we have efficiently utilized those properties to achieve hardware reduction and a novel parallel Architecture.

In Medical Image Processing System, characterization and recognization of suspended minute particles in cellular fluid is an important problem. The two key steps involved are up-sampling of the region of interest (RoI) and edge detection [8]. In our case, the target RoI's are already blurred due to the presence of surrounding cellular fluids. So Upsampling of these RoIs makes them more blurred and as a consequence, the subsequent edge detection becomes a futile exercise. Henceforth arises the requirement of an interpolation algorithm, which preserves some high pass contents. The B-Spline interpolation as can be shown in Z domain satisfies the above-mentioned constraint [1].In past, few papers like [2] dealt with the implementation of B-Spline hardware. However they failed to exploit some important properties of basis functions like periodicity for Data Reuse as will be shown in the next section. Here we have efficiently utilized those properties to

achieve hardware reduction and a novel parallel architecture.

A reprogrammable and variable order B-Spline generator and interpolator circuit is also provided herewith. Finally some suitable modifications are proposed in our architecture to reduce the power consumption as in [3].

2 SYSTEM ANALYSES

Existing system

An image may be defined as a two dimensional function f(x, y) [7] where x and y are spatial (plane) co-ordinates and the amplitude of f at any pair o

The VHDL language can be [6] regarded as an Integrated Amalgamation of the following languages such as sequential language, concurrent language, net-list language, Timing language, waveform generation language. f co-ordinates (x, y) is called the intensity or gray level of the image at that point. Gray level is used to describe monochromatic intensity [7] because it ranges from black to grays and finally to white. Digital Image is composed of a finite number of elements, each of which has a particular location and value .These elements are referred to as picture elements, image elements, pels or pixels.

Region of Interest (RoI)

RoI's are used primarily to isolate an area for processing. This is done to highlight that area and differentiate it from the rest of the image. In our case, the target RoI's are already blurred due to the presence of surrounding cellular fluid. So we go for edge detection.

Edge Detection

Edge is a set of connected pixels that lie on the boundary between two regions. Edge detection [8] is the most common approach for detecting meaningful discontinuities in gray level. Edge detection is the first step in recovering information from images.

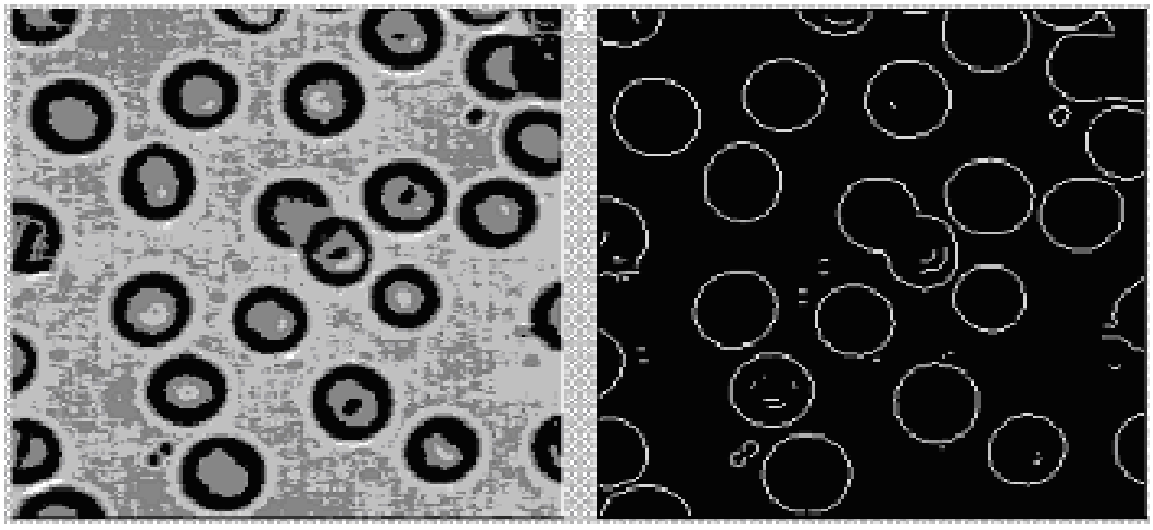

Figure 1 original blood image

Figure 2 Edge map

After edge detection, each edge pixel is assigned to the foreground value (255) while non edge pixels are assigned to the background value (0).In the

processed image, edges will appear as white lines on a black background.

Drawback

By Sobel's edge detection very minute particles suspended in the fluid cannot be detected.

3 B-SPLINE THEORY

From a mathematical point of view, a curve generated by using the vertices of defining polygon is dependent on some interpolation of approximation scheme to establish the relationship between the curve and the polygon. This scheme is provided by the choice of basis function. Two characteristics of the Bernstein basis, however, limit the flexibility of the resulting curves. First the number of specified polygon vertices [5] fixes the order of the resulting polynomial which defines the curve. For example. A cubic curve must be defined by a polygon with four vertices and three spans. A polygon with six vertices always produces a fifth-degree curve. The only way to reduce the degree of the curve is to reduce the number of vertices, and conversely the only way to increase the degree of the curve is to increase the number of vertices.

The second limiting characteristic is due to the global nature of the Bernstein basis. This means that the value of the blending function Jn, i (t) is nonzero for all parameter values over the entire curve. Since any point on a Bezier curve is a result of blending the values of all defining vertices, a change in one vertex is felt throughout the entire curve. This eliminates the ability to produce a local change within a curve.

Flexibility of B-Spline basis function

1 Type of the knot vector can be changed and hence the basis functions.
  Order of the basis function k can be changed.
2 The number and position of the defining polygon vertices cab be changed.
3 Multiple polygon vertices can be used.
4 Multiple knot values in the knot vector can be used.
5 With increasing multiple vertices (k-1) sharp corner is created.
6 By increasing the order of the curve the smoothness of the curve can be increased as shown in figure below.

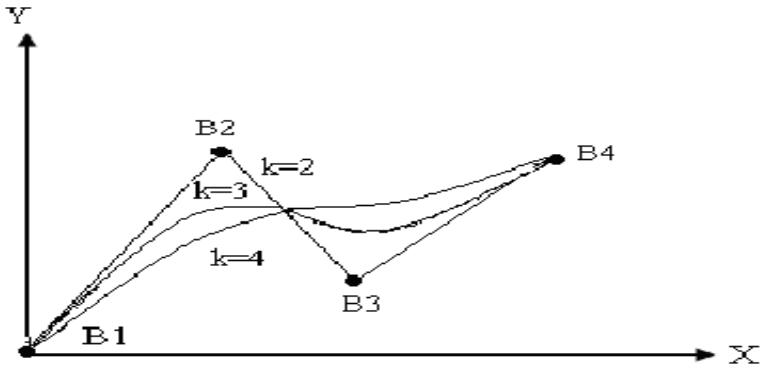

Figure 3 Curve showing the effect of change in order

4. IMPLEMENTATION DETAILS

Proposed system is the B-spline curve. In this system [1] curve before apply the mathematical calculation we have to apply some techniques given below to get efficient implementation of b-spline curve.

1. FIR FILTER
2. SMOOTHING
3. INTERPOLATION

FIR filter

B-spline interpolation correctly applied does not result in a loss of image

SMOOTHING

Smoothing filters are used for blurring and for noise reduction blurring is used in preprocessing steps such as removal of small gaps in lines or curves. Noise reduction can be accomplished by blurring with a linear filter and also nonlinear filtering.

Regardless of the type of linear filter used, however the basic approach is to sum products between the mask co-efficient and the intensities of the pixel under the mask at a specific location in the image. Denoting the gray levels of pixels under the mask at any location by z_1, z_2,…..z_9 the response of a linear mask is $R=w_1z_1+w_2z_2+…….w_9z_9$. If the center of the mask is at location (x, y) in the image, the gray level of the pixel located at (x, y) replaced by R

like

W_1	W_2	W_3
W_4	W_5	W_6
	W_8	W_9

Figure 4 3*3 Mask Matrix

Interpolation

Interpolation is the process of increasing the resolution of an image. The process is required when there is not enough resolution in an image to accomplish the reproduction needed. Image interpolation is required for resolution conversion to adapt to the characteristics of a particular display device.

Sometimes an image can't be reproduced at the size needed. It lacks the needed resolution (full size) or has too much for practical use. In these cases, the computer can be called upon to modify the resolution of an image, reducing its resolution or increasing it.

When an image is reduced in resolution, the process is called decimation. The computer scans thro a line of pixels, casting off unnecessary information by averaging together pairs of pixels, or groups of pixels to the average of the original group.

There are various types of interpolations

Zero order (nearest neighbour)

First order (bilinear)

There are two steps involved in interpolation

1. New pixel location
2. Assignment of gray levels to that new pixel

A slightly more sophisticated way of accomplishing gray-level assignments is bilinear interpolation using the four nearest neighbor of a point. Let

resolution and it can be performed in a very efficient manner.

The efficient implementation of an interpolator, which is realized by first inserting Zeros between the samples of x(n) and the filtering the resulting sequence. In the filter design process, we determine the co-efficient of a causal FIR filter.

$$Y(n) = \sum_{K=0}^{M-1} b_K x(n-k) \quad \text{------} \quad (1)$$

$$H(Z) = \sum b_k z^{-k} \quad \text{------} \quad (2)$$

(x', y') denote the co-ordinates of a point in the zoomed image (think of it as a point on the grid described previously), and let v (x', y') denote the gray-level assigned to it .For bilinear interpolation, the assigned gray level is given by

$$V(x', y') = ax'+by'+cx'y'+d$$

Where four coefficients are determined from the four equations in four unknowns that can be written using the four nearest neighbors of point (x', y').

5 B-SPLINE CURVES

The convex hull property of B-spline [9] curves is stronger than that for Bezier curves. For a B-spline curve of order k (degree k - 1) a point on the curve lies within the convex hull k neighboring points. Thus, all points on taking k successive defining polygon vertices. In Fig.5 where the convex hulls shown shaded, illustrates this effect for different values of k. notice in particular that for k = 2 the convex hull is just the defining polygon itself. Hence, the B-spline curve is also just the defining polygon.

Using the convex hull property it is easy to see that if all the defining polygon vertices are collinear, then the resulting B-spline curve is a straight line for all orders k. Further, if l colinear polygon vertices occur in a noncolinear defining polygon, then the straight progtions of the defining curve (if any) start and end at least k - 2 spans from the beginning and end of the series of colinear polygon vertices is completely contained within a noncolinear defining polygon, then the number of colinear curve spans is at least l - k + 1. Figure 5 illustrates these results.

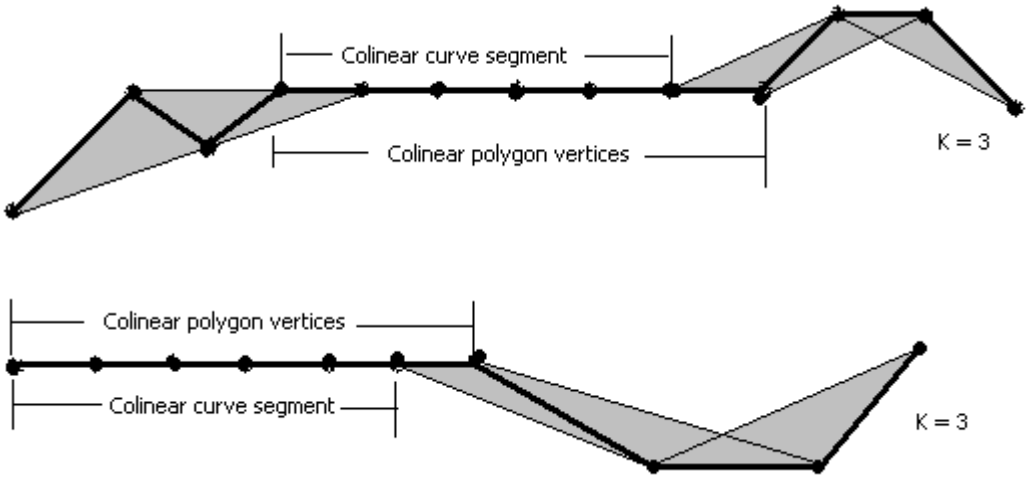

Figure 5 Convex hull properties of B-spline curves for collinear

Polygon vertices

B-Spline convex hull properties for collinear curve segments clearly show that the choice of knot vector has a significant influence on the B-spline basis functions $N_{i,k}$ (t) and hence on the resulting B-spline curve. The only requirement for a knot vector is that it satisfies the relation $x_i <= x_{i+1}$; i.e., it is a monotonically increasing series of real numbers.

Knot Vectors

Fundamentally three types of knot vector are used: uniform, open uniform (or open) and nonuniform.

In a uniform knot vector, individual knot values are evenly spaced. Examples

[-0.2 -0.1 0 0.1 0.2]

In practice, uniform knot vectors generally begin at zero and are incremented by 1 to some maximum value or are normalized[1] in the range between 0 and 1, i.e., equal decimal intervals, e.g.,

[0 0.25 0.5 0.75 1.0]

for a given order k, uniform knot vectors yield periodic uniform basis functions for which[11]

$N_{i,k}(t) = N_{i-1,k}(t - 1) = N_{i+1,k}(t+1)$----- (3)

Thus each basis function is a translate of the other.

An open uniform knot vector has multiplicity of knot values at he ends equal to the order k of the B-spline basis function. Internal knot values are evenly spaced. Some examples using integer increments are

K= 2 [0 0 1 2 3 4 4]

K=3 [0 0 0 1 2 3 3 3]

K=4 [0 0 0 0 1 2 2 2 2]

Formally, an open uniform knot vector is given by

$x_i = 0$ $1 <= i <= k$

$x_i = i - k$ $k + 1 <= i <= n+1$

$x_i = n - k + 2$ $n + 2 <= i <= n + k + 1$

Eg: For k =2, n=4

(0 0 1 2 3 4 4)

For k =3, n=4

(0 0 0 1 2 3 3 3)

The resulting open uniform basis functions yield curves that behave most nearly like Bezier curves. In fact, when the number of defining polygon vertices is equal to the order of the B-spline basis and an open uniform knot vector is used, the B-spline basis reduces to the Bernstein basis.

Non uniform knot vectors may have either unequally spaced and / or multiple internal knot values. They may be periodic or open. Examples are

[0 0 0 1 1 2 2 2]

[0 1 2 2 3 4]

[0 0.28 0.5 0.72 1]

B-Spline Theory

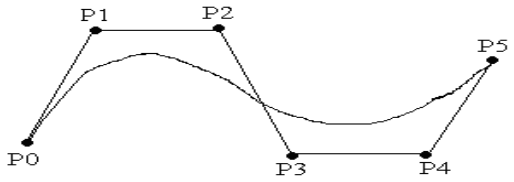

Figure 6 B-Spline curves

A B-spline curve as in [5] can be mathematically written as

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t) \quad (5.2) \quad t_{min} <= t <= t_{max}, 2 <= k <= n + 1$$

Where P (t)[10] is position vector along the curve as a function of parameter (t); Bi is the Position vectors of (n+1) defining polygon vertices.

Ni, k (t) are the normalized basis function for the k-th order curve and is given by

$$N_{i,1}(t) = 1 \quad \text{for} \quad x_i < t < x_{i+1}$$

$$N_{i,1}(t) = 0 \quad \text{otherwise} \quad \text{--------} \quad (4)$$

$$N_{i,k}(t) = \frac{(t - x(i)) * N_{i,k-1}(t)}{x(i+k-1) - x(i)} + \frac{(x(i+k) - t) * N_{i+1,k-1}(t)}{x(i+k) - x(i+1)} \quad (5)$$

Here xi are the elements of 'k' knot vectors and

$$k - 1 <= t <= n + 1$$

The parameter t varies from tmin to t_{max} along the curve P (t). The convention 0/0 = 0 is adopted.

Formally a B-spline curve is defined as a polynomial spline function of order k (degree k - 1) since it satisfies the following two conditions:

The function P (t) is a polynomial of degree k - 1 on each interval x_{i+1}.

P (t) and its derivatives of order 1,2,…., k - 2 are all continuous over the entire curve. The maximum order of the curve is equal to the number of defining polygon vertices.

Serial computation

Fig.6 (Using the notation that the convergence of two edges means a summation operation) Weight of an edge <Ni,k-1,Ni,k> is equal to

$$W1 = a = (t - xi) / (xi+k-1 - xi) \quad (6)$$

Weight of an edge <Ni+1, k-1, Ni, k> is equal to

$$W2 = b = (xi+k - t) / (xi+k - xi+1) \quad (7)$$

A large number of arithmetic processors and storage elements will be required if one tries to implement the above mentioned dependence graph directly. Additionally one also needs to calculate N2, 5, N3, 5,....N (n+1, 5). The repetition of the whole process is required[1] whenever value of't' changes. Hence the system will be quite inefficient if implemented in a serial or sequential fashion

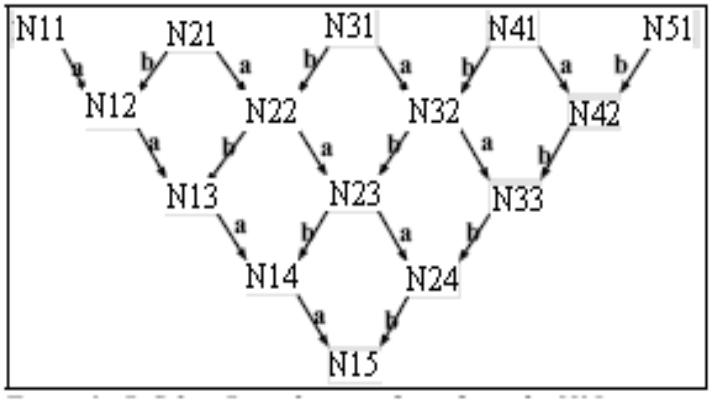

Figure 7 B-Spline Basis function dependence for N15

Parallel computation

For Parallel computation we exploit some special properties of the B-Spline basis functions

Property 1:-

Periodicity of Basis Function.

$$N_{i,k}(t) = N_{i-1,k}(t-1) = N_{i+1,k}(t+1) \quad \text{---------} \quad (8)$$

Using the above property, in general, for calculation of $N_{i,k}(t+i-1)$, only $N_{1,k}(t)$ is required where $0 <= t <= 1$. Hence if we need to draw a curve within the range of $x <= t < x+p$, the whole job can be divided into several smaller job segments given by,

$$t = x \rightarrow x+1;\ t = x+1 \rightarrow x+2 \ldots t = x+p-1 \rightarrow x+p; \quad \text{-----} \quad (9)$$

Each of the job-segments will run in a parallel fashion. The process of parallel run is tabulated in Fig.10. It is to be noted that the valid range of 't' is given by $k-1 <= t <= n+1$ ---------- (10)

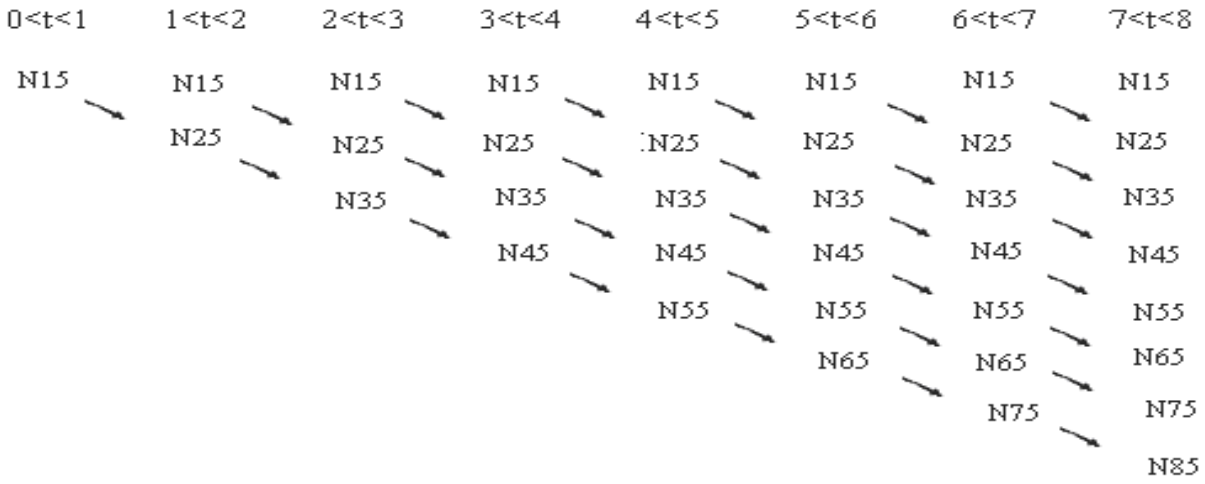

Figure 10 Job divisions in Parallel Processing

Property 2:-

$$\sum N_{i,k}(t) = 1 \text{ for } k-1 <= t <= n+1 \quad \text{-------} \quad (11)$$

Using the above property [10] we see that in the valid curve drawing parameter range of 't'

$$N_{1,5} = 1 - \sum N_{i,5} \quad \text{where } i=2, 3, 4, 5.....8 \text{ ---------} \quad (12)$$

In Fig.11 a dependence graph for calculation of $N_{1,5}$ in various patches of parameter 't' has been shown.

Here column (i) calculates $N_{1,5}$ for $i-1 < t < i$.

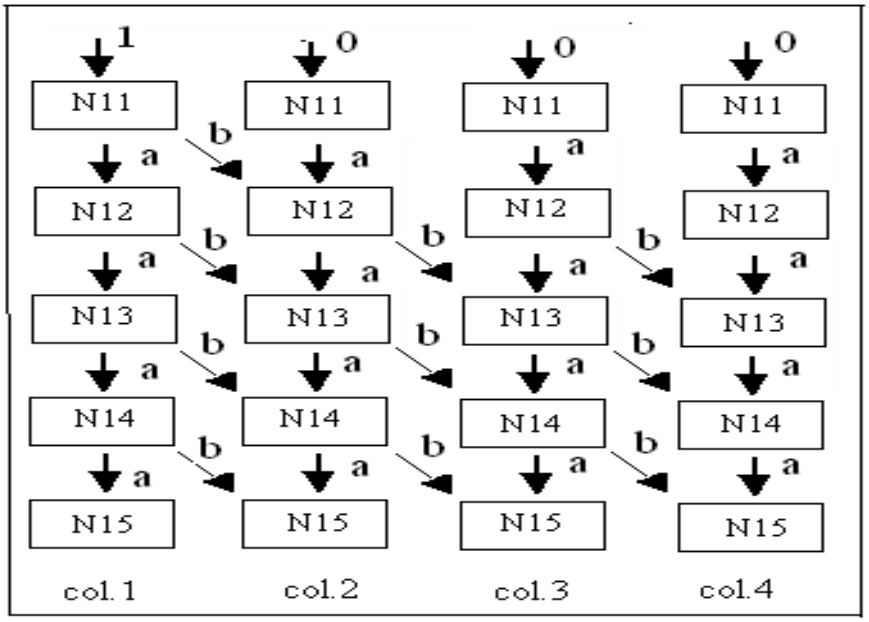

Figure 11 Dependence graph for Parallel Architecture System

From Eqn.(6), (7) we observe that the calculation of N1,j(t) needs N1,j-1(t) and N1+1,j-1(t) .Using Property 1, we get N1+1,j-1(t) = N1,j-1(t-1); Hence in the graph, N1,j-1 from each column goes to N1,j of the next column through an edge with weight 'b' as given in Eqn(6)

6 DESIGN OF THE PARALLEL ARCHITECTURE

Fig.12 shows the proposed parallel architecture for the calculation of the basis function for a 5^{th} order curve. Thus generating the curve within the valid parameter range of t (4<=t<=n+1=8) we require to increment't' from 4 to 5 only. Column-i is fed by (td + i -1), td being the decimal portion t. In processor architecture of Fig.12, N1, 5 of column-i is fed the value of N1, 5(ti) where i-1<=t_i<i. Fig.12 depicts the hardware required for the calculation of N1, 5(ti) within the range of 4<= t_i < 8 as stated in Eqn. (2.12)

Let the value of't' lie between 6 and 7. Considering the direct implementation of Fig.7 we find that 20 multipliers are required for calculating N15. Similarly we need to calculate N25, N35…. N75. The total multipliers required henceforth are 20+ (7-1) x8 = 68; In our proposed parallel architecture as in Fig.11 and 12 we find that all N15, N25,…. N75 will be calculated in single run. Hence 26 multipliers will suffice our purpose.

Let us calculate all N15 (t)'s within a range of 4<=t<8 with an increment of 0.1. The direct implementation of Fig.7 requires 20 multipliers to be used 40 times or 800 multipliers equivalently. The proposed parallel architecture of Figure 12 would require 10 runs only. Hence it would require 260 multipliers only.

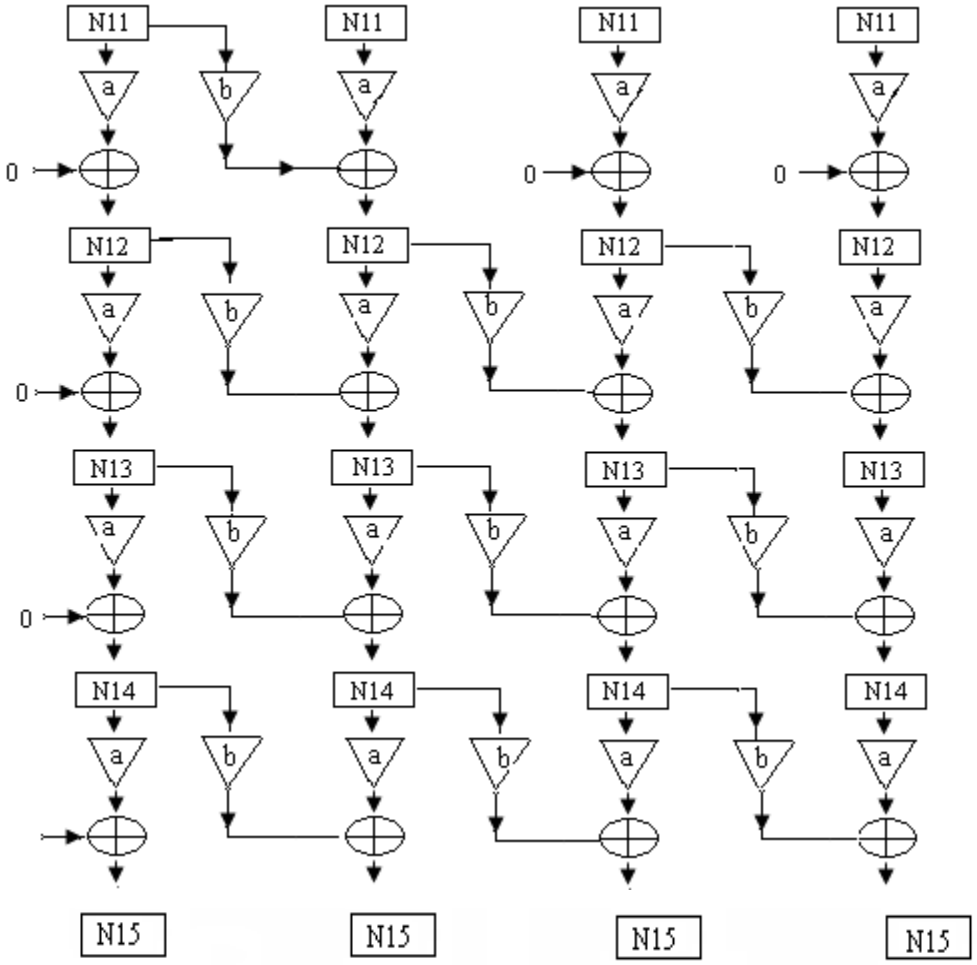

Figure 12 Architecture for parallel processing

7 HARDWARE REDUCTION

For further reduction of hardware in B-Spline function generator, we can apply some DSP architecture optimization methods. One such efficient method is folding of the previously described architecture.

There are two kind of folding architecture

1. Column folding
2. Row folding

Column folding

The column folded architecture is shown in Figure 2.10 The AND gate, whose o/p is being fed into the latch's clock i/p, is basically a combinatorial logic gate which gives output 1 when up counter o/p is binary 3.

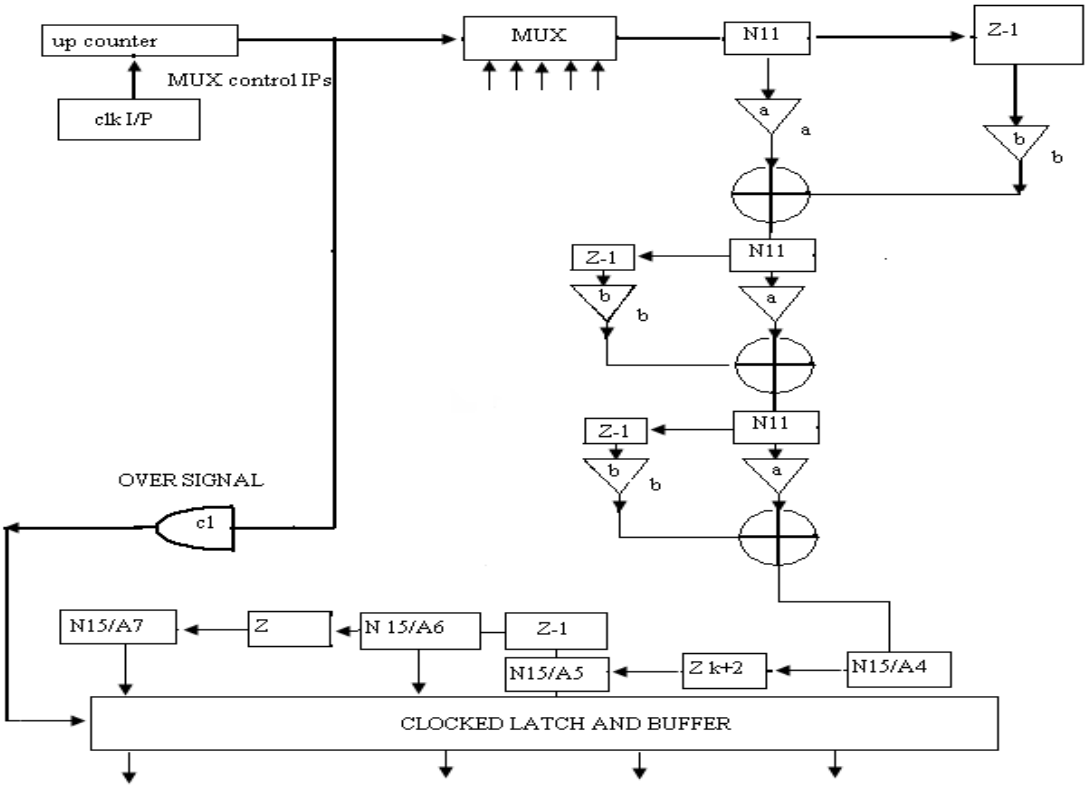

Figure 13 Column Folded Circuit for 5th order

Row folding

In Figure 2.11 we have performed row-wise folding of Figure 2.9 Here multiplexer and de-multiplexers are used for purpose of switching. The AND Gate, with mark C1, is basically a combinatorial logic gate which gives output 1 when up counter o/p is binary 3.

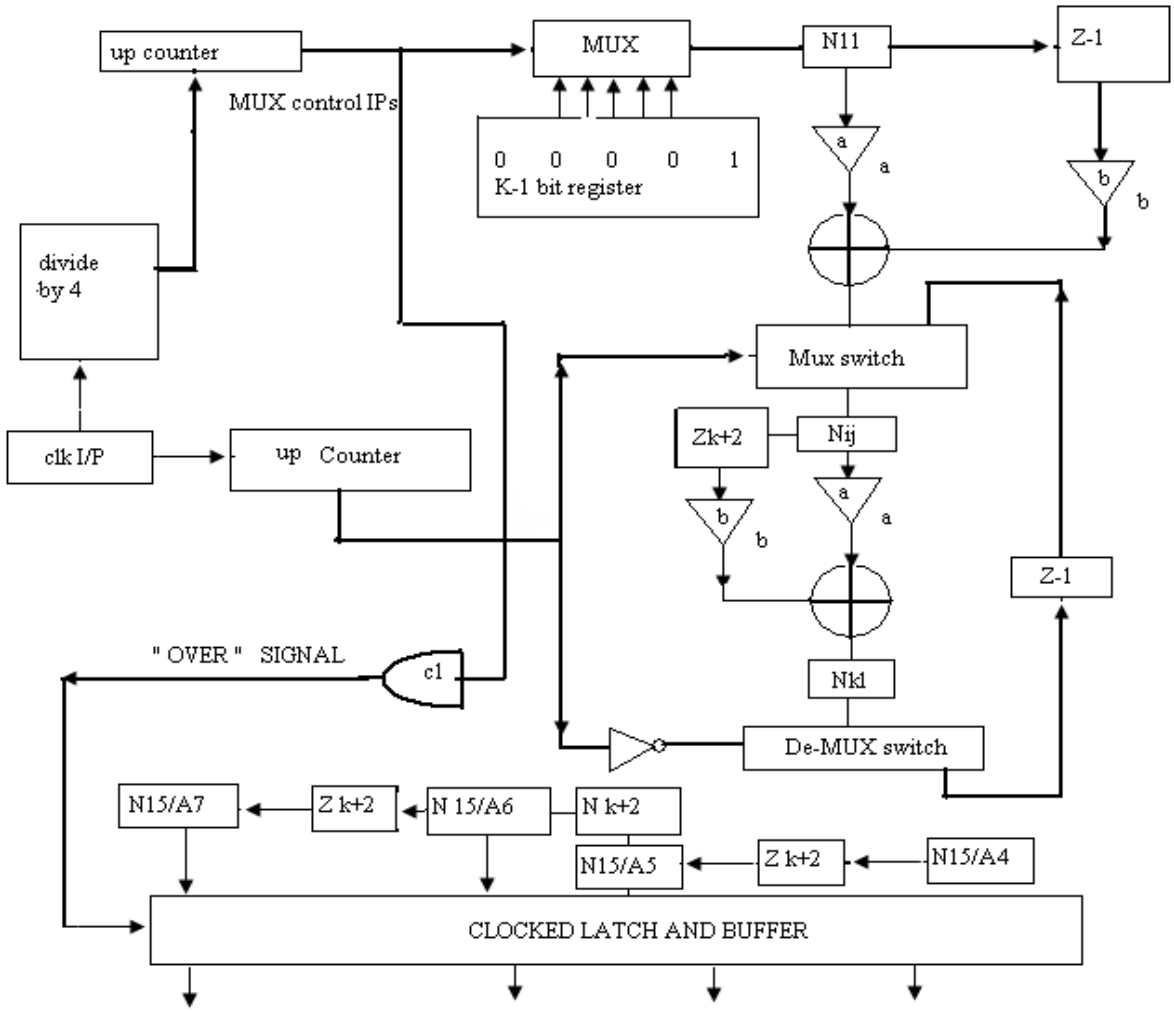

Figure 14 Row and Column Folded circuit for 5th order

Variable order reprogrammable circuit

In Figure 15, it is evident that instead of Z-3 if we utilize a delay of Z-K+2, where K is the variable order of the circuit, then we get a reprogrammable circuit. We also require variable MOD counters. Ring counter suites best, as it requires minimal decoding hardware. To make variable mod ring counters; we require a variable length shift register.

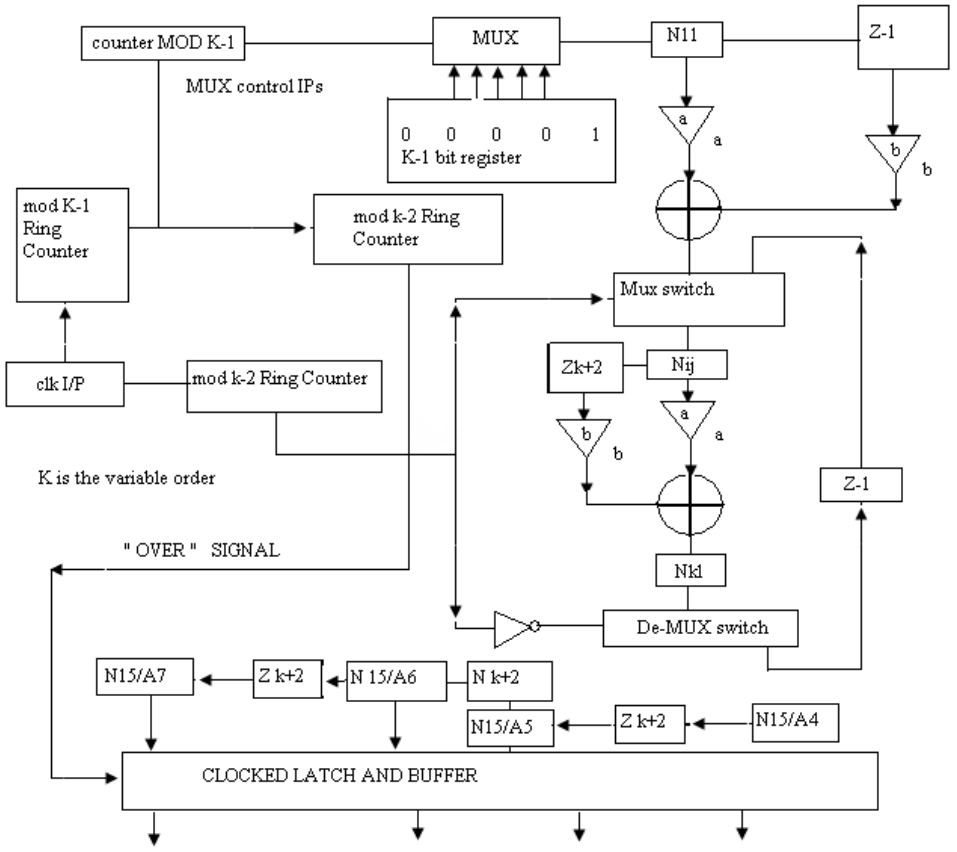

Figure 15 variable order B-Spline basis function generator

8 FINITE WORD LENGTH EFFECT

Case 1. Sequential Implementation

Consider the basis function dependence graph of Figure 7 The weighing operation involves multiplication [11]operation after which a round-off operation is necessary to incorporate the output result in finite word length.

For ease of mathematical analysis of the errors, we model the fixed point multiplier as in Fig.16.

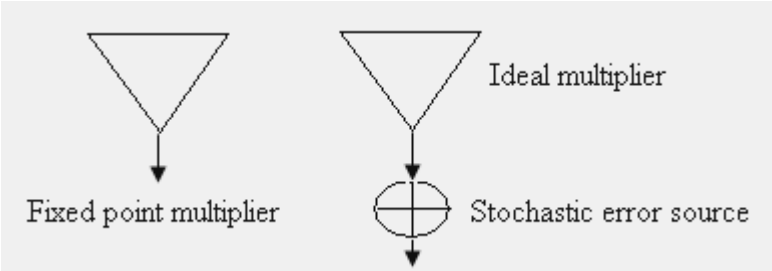

Figure16 Stochastic multiplier models

Case 2. Parallel Implementation

As shown in Fig.11 we first calculate N1, 5 for each of the four columns. Following Fig.10 (Table-1), these N1, 5's act as various N5, 5's within a range of k-1<=t<=n+1. The errors propagated in the N5, 5 are same as that of N1, 5's in various columns without any further multiplicative factors, thus reducing the errors in N5,5 to a large extent.

9 RESULTS AND DISCUSSION

The recognization of tiny interpolated object using B-Spline is done here. Initially we have to get any image, for which we are to finding edges using B-Spline, that figure may be of with noise or without noise. To get clear image we have to do the process of filtering, smoothing, interpolation in that fig then applying B-spline theory. Software process in viz Matlab and VHDL are done. Finally getting the results according to the vertices value.

The result is some what smooth. When k=3, m=16 However for better smoothening than k=1, m=17. The vertices and order value is choosen accordingly.

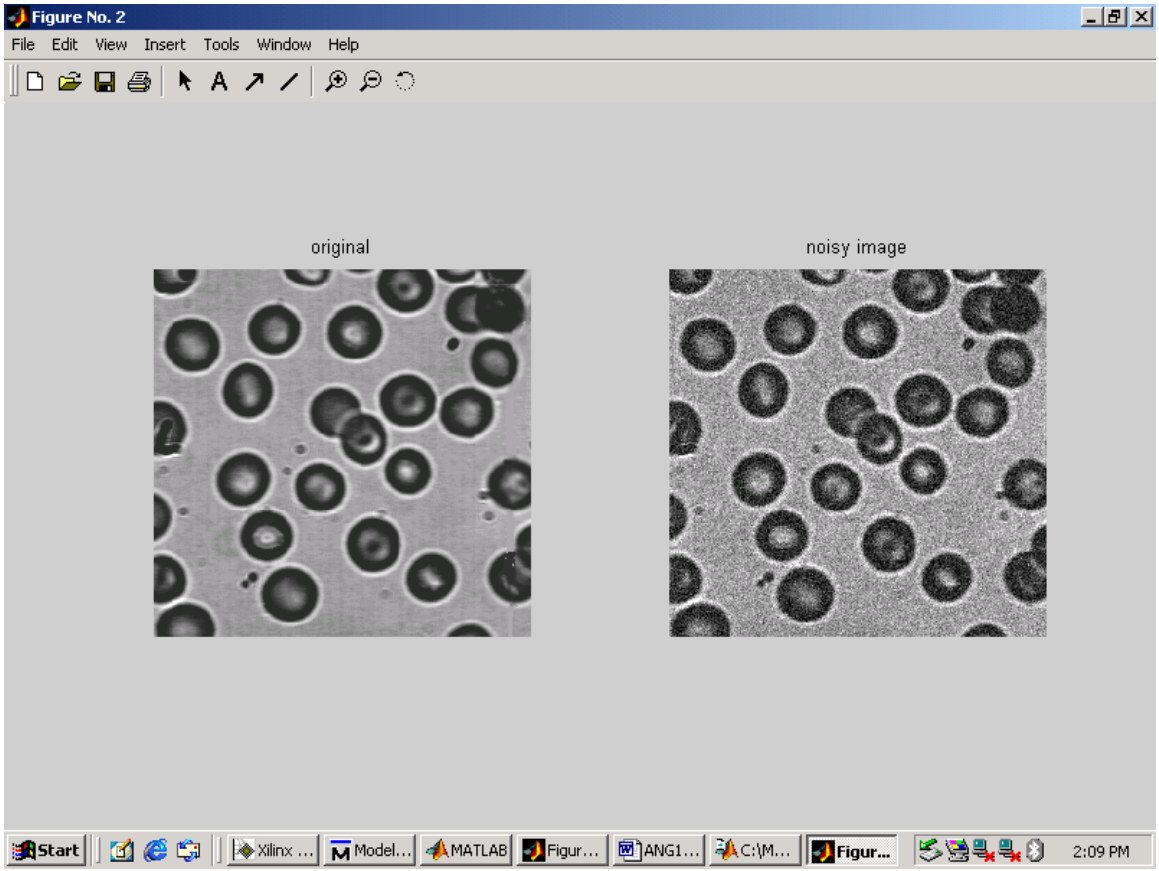

Figure17 Input Images

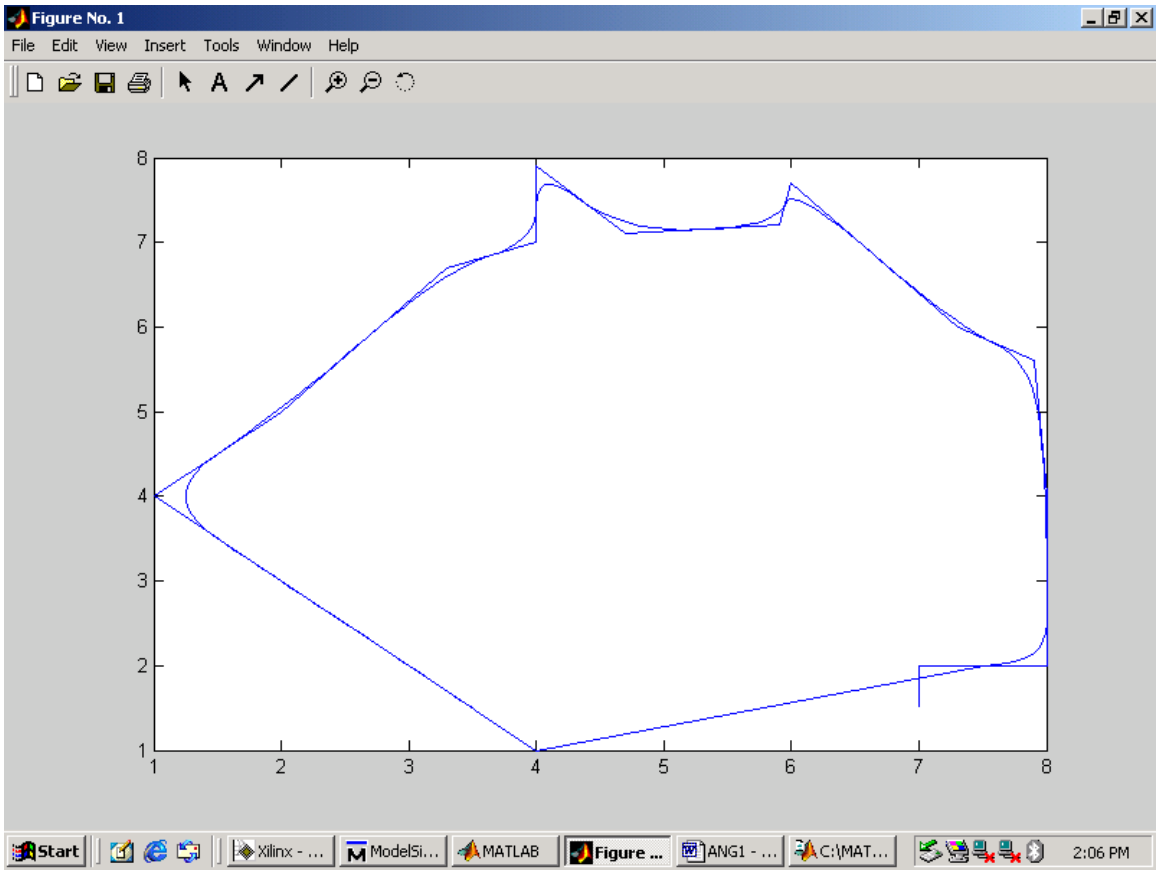

Figure 18 output figure for the value of K=1 and M=17

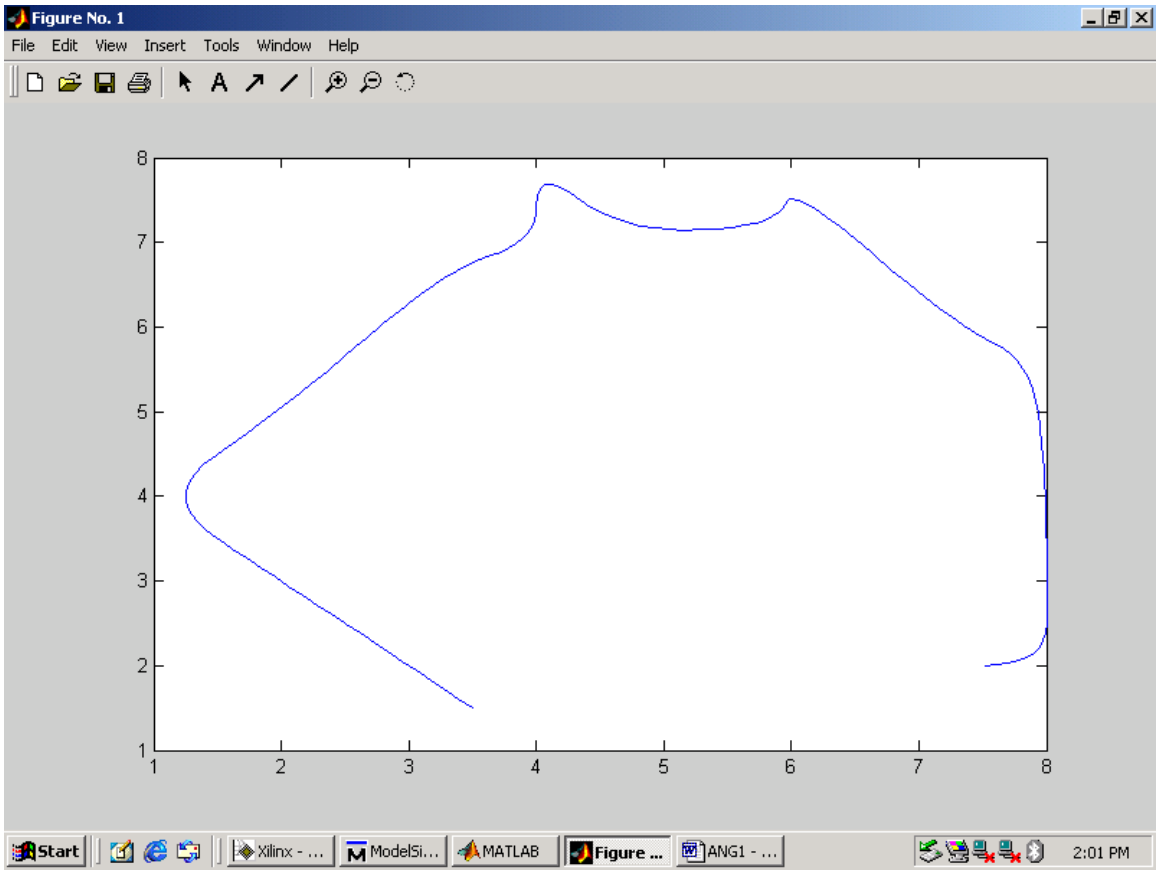

Figure 19 output figure for the value of K=3 and M=16

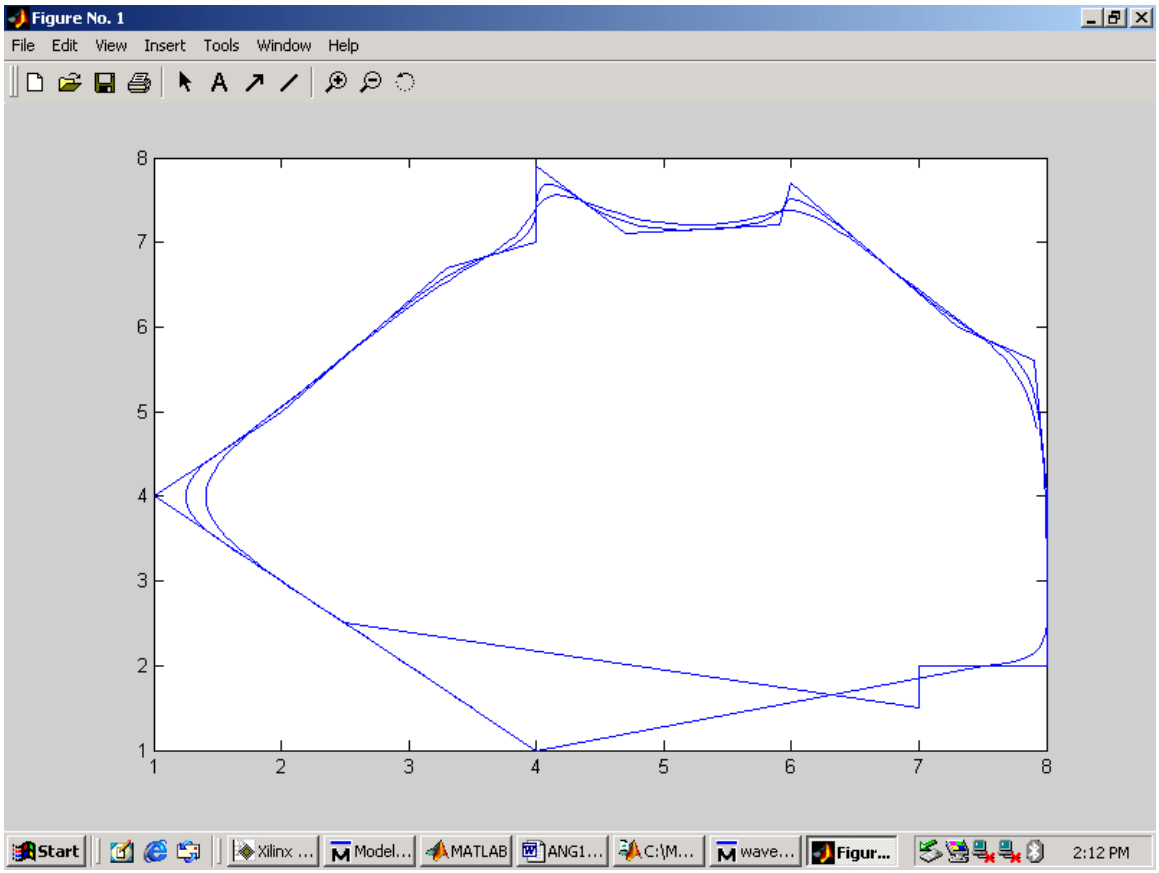

Figure 20 output figure for the value of K=5 and M=14

CONCLUSION

Hardware minimization and device minimization is the key essence of a modern VLSI design. In this project we have provided two different extreme cases, fully parallel architecture and fully compact architecture. It is indeed a matter of further research to have a suitable tradeoff between speed and area upon the requirement user.

REFERENCES

1. Aldroubi, A.; Eden, M.: Unser, M., "Fast B-spline transforms for continuous image representation and interpolation", Pattern Analysis and Machine Intelligence, IEEE Transactions on, Volume: 13, Issue: 3, March 1991

2. Arijit De, S.Mondal, P.K.Biswas, "A Fixed Point Parallel Architecture for generation of B-spline patches", Proceedings Conference on Distributed Processing and Networking'04, Kharagpur ,June 11-13, 2004 ,pp.231-237.

3. .M.P. Beddoes and T.K.Chu, " Direct Sample Interpolation (DSI) Speech

Synthesis: An Interpolation Technique for Digital Speech Data Compression and Speech Synthesis," IEEE Transactions on Accounts, Speech and Signal Processing, vol.ASSP-30, pp.825-831, 1982.

4. Chandrakasan, A.P.; Sheng, S.; Brodersen, R.W., "Low power CMOS digital design", Solid-State Circuits, IEEE Journal of, Volume: 27, Issue: 4, April 1992

5. M.Unser, A.Aldroubi, and M.Eden, "B-spline signal processing – Part I: Theory," IEEE Trans.Siganl Processing, vol.41, pp.821-833, 1993.

6. Neil H. E. Weste and Kamran Eshraghian, Principles of CMOS VLSI Design, Pearson Education ASIA, 2nd edition,2000

7. Anil K.Jain," Fundamentals of Digital Image Processing" , PHI 1995.

8. R.Gonzalaz and P.Wintz, "Digital Image Processing ", Addison Weseley nd Ed , 1987.

9. Y. Liu, H.Yang, and W. Wang, "Reconstruction B-Spline Curves from Point Clouds – A Tangential Flow Approach Using Least Squares Minimization," Proceedings of the International Conference on Shape Modeling and Applications," 2005.

10. D. Rogers, Mathematical elements of computer graphics, McGrawhill International Edition.

11. K.Toraichi, K. Katagishi and R. Mori, "A Fast B.Spline Trasform and its Applications,"ICASSP, vol.86, pp.301-304, 1986.