Journal of Circuits, Systems, and Computers

NoC based Implementation of Free Form Deformations in Medical Imaging Registration

Uzma Mushtaq and Osman Hasan

School of Electrical Engineering and Computer Science
National University of Sciences and Technology (NUST)
Sector H-12, Islamabad, Pakistan
{*09mseeumushtaq,osman.hasan*}*@seecs.nust.edu.pk*

Falah Awwad

College of Engineering,
UAE University, Al-Ain, UAE
f_awwad@uaeu.ac.ae

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

These days, computer based image registration techniques are increasingly being used in the area of medical imaging as they offer significant benefits for aligning different images together and for visualizing their combined images. However, these techniques require an enormous amount of computation time due to the high resolution and complex nature of the medical images. We propose to alleviate this problem by using a dedicated Network-on-Chip (NoC) based hardware platform for image registration. This paper describes a novel technique for FPGA implementation of the B-Spline based Free Form Deformation (FFD) algorithm, i.e., a widely used algorithm for modeling geometric shapes in a computerized environment. For performance enhancement, we have utilized a a lightweight circuit-switched NoC architecture, which is adaptable to most FPGAs. The design description is captured in the Verilog language and implemented using the Xilinx XC2v6000 device at 37 MHz. The proposed design is parameterizable at the compile time and supports a wide range of the image resolutions and computational precisions. The experimental results have shown a significant improvement in performance when compared with the other existing hardware implementations of the B-Spline based FFD algorithm.

Keywords: FPGA; Medical Imaging; Network-on-Chip Architectures

1. Introduction

Traditionally, in the medical domain, multiple radiological images of a patient are acquired, printed and then analyzed by viewing them on a light box. The computer based image registration techniques offer significant benefits for aligning different images together and for visualizing their combined images and thus are increasingly being used in medicine. For example, it is quite difficult to localize tumors using

CT and MR scans based images because the contrast between the tumor and its surrounding tissues is of very low intensity [1]. However, image registration has been shown to enhance the tumor detection significantly [2].

Medical image registration methods are primarily based on iterative algorithms that tend to minimize some cost or energy factor, which is usually defined in terms of the difference of geometry or intensity between images. Thus, the efficiency of image registration is directly dependent on the performance of the underlying algorithms. A significant amount of research has been conducted to explore efficient algorithms for medical image registration. Free Form Deformation (FFD) [1] based algorithms are most commonly used in medical imagining mainly because they support the modeling of geometrical shapes in a computerized environment. For example, the medical imaging utilization of a non-rigid registration algorithm based on the FFD and modeled by B-splines is explained in [3]. Similarly, another algorithm for the non-rigid registration of 3-D breast MRI is also investigated [4].

One of the common problems associated with the image registration algorithms is their enormous computation complexity due to the high resolution and complex nature of the medical images. For example, a 3D image of 256x256x64 voxels is processed in 15-30 minutes using the FFD algorithm on a Sun Ultra 10 workstation [5]. In order to optimize the performance of FFD algorithms, dedicated hardware platforms have been proposed to be used for executing non-rigid image registration algorithms [2,6]. One of the most recent works, oriented towards this direction of research is the reconfigurable hardware based FPGAs implementation to compute the B-spline based FFDs for medical imaging [5]. This paper contributes towards further performance enhancement of the B-spline based FFD algorithm by using a Network-on-Chip (NoC) [7,8,9] based hardware implementation for this purpose. NoC is basically a computing system consisting of several interconnected and concurrently running processors. Scalability, design-flow parallelization, and reusability are the main benefits of NoC based implementations. Thus, due to the inherent nature of NoC architectures, the proposed approach of using NoC architecture is expected to improve upon the performance of the FFD algorithm based on B-Spline.

We propose to use a simple circuit-switched architecture called programmable NoC (PNoC) [10]. PNoC is a very flexible and lightweight architecture for FPGA based systems. PNoC uses a modular design that facilities the usage of standard interfaces and IPs [11]. Higher communication bandwidth and better scalability are the foremost merits of PNoC. The main contribution of this paper is the implementation details of the B-Spline based FFD algorithm using PNoC. The analysis results of our implementation are compared with the reconfigurable FPGA based approach of the same algorithm [5], and it has been observed experimentally that the proposed approach led to a noticeable performance increase and cost reduction. To the best of our knowledge, this is the first time that a NoC based architecture has been proposed to be used in the context of medical image registration applications.

The rest of the paper is organized as follows: Section 2 provides some preliminary information regarding the NoC architectures and the B-Spline FFD algorithm. In

Section 3, we describe the proposed NoC based implementation of the B-Spline based FFD algorithm. The results and comparisons with the reconfigurable FPGA based approach are presented in Section 4. Finally, Section 5 concludes the paper.

2. Preliminaries

To facilitate the understanding of the rest of the paper, we provide some fundamentals regarding NoC, PNOC architecture and the B-Spline FFD algorithm in this section.

2.1. *Network-on-Chips*

The basic ingredients of a NoC architecture, depicted in Figure 1, include the processing elements, connection topology, routing technique, switches, and programming model. There are various connection topologies from the communication perspective. Torus, octagon, mesh, ring, and irregular connection networks are some of the communication topologies [13]. However, it has been shown that the 2-D mesh architecture is both easy to implement and provides the lowest latency [14]. Different

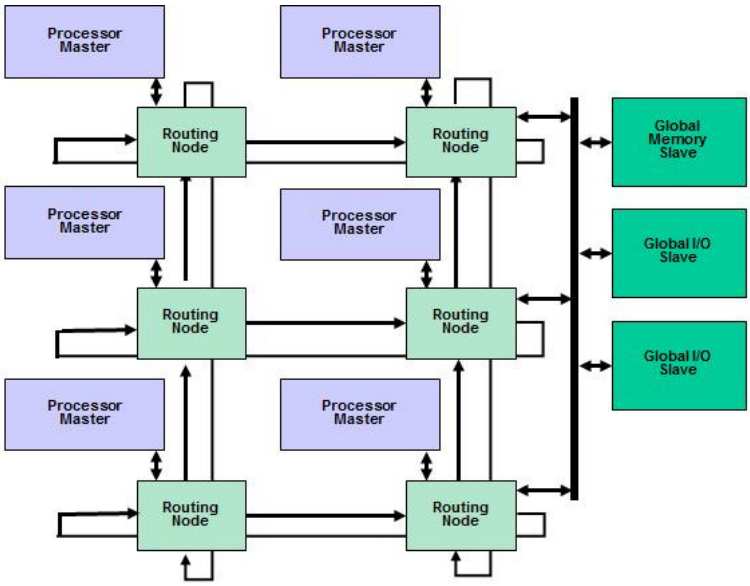

Fig. 1. NoC Architecture

architectures for NoC routing have been proposed but the most widely used ones are packet switched and circuit switched. The packet switched routing architecture [15] requires some amount of buffering. Buffers are provided at input, output or

both input and output. In the circuit switched architecture [16], there is a dedicated channel for the data flow so no buffering or queuing is required.

2.2. *PNOC Architecture*

In this paper, our focus is on a NoC based implementation of a FFD algorithm and thus for this purpose, our requirement is to use a flexible and lightweight NoC architecture. The lightweight circuit-switched NoC for FPGA based systems, described by Hilton et al [10], fulfills our needs. In PNoC, the network consists of subnets, such that each subnet has a router and a bunch of network nodes shown in Figure 2.

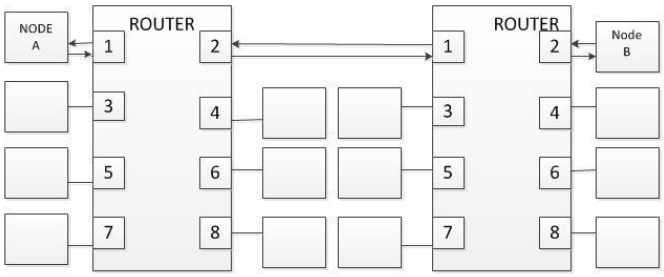

Fig. 2. PNoC Network Topology [9]

The circuit switching between the nodes is performed by the router and each node is connected to a router. A dedicated connection is established using a light handshaking mechanism for the data exchange and connection removal. The connection is established when master node A sends the request signal along with the address of the target node to the router. The second router sends the grant signal to first router that port B is available and the connection is established.

A dedicated connection path is used for data transfer so no acknowledge signal is required. Data transaction can occur on successive clock cycles if master receiver is low. The read and write requests can be pipelined. A CPU is connected to the PNoC like any other module. The interfacing circuit constitutes FIFO's and FSM to communicate with the router. The router is the main component of the PNoC. The router includes the routing table, queue, and switch box. Another part of the PNoC is the buffer which is a parameterizable feature. Buffer is necessary in two cases. Firstly, if nodes and routers are running at different clock rates, and secondly, when there is a difference between the transmitting and receiving rate.

The PNoC has been implemented using JHDL [10], which is not a commonly used HDL. In order to facilitate broad usage of PNoC, we implemented its design in Verilog [17]. We use this Verilog implementation to develop the NoC based

implementation of the B-Spline FFD algorithm.

2.3. *B-Spline based FFD Algorithm*

The B-Spline based FFD algorithm is considered to be one of the most powerful techniques for modeling 3-D deformable objects in the domain of non-rigid image registration [4], which is a special kind of image registration used specifically for images with nonlinear geometric differences. The main motivation behind the optimization of this algorithm in this paper is the increasing utilization of non-rigid registration for the analysis of huge and complex brain images.

For non-rigid image registration, a combined transformation (T) using both global and local transformation is used [5].

$$T(X;Y;Z) = T_{global}(X;Y;Z) + T_{local}(X;Y;Z) \tag{1}$$

In the case of the B-Spline FFD, the image volume is defined as $\Omega = \{(x,y,z)|0 \leq x \leq X, 0 \leq y \leq Y, 0 \leq z \leq Z\}$. Thus, the FFD can be described as the product of 3 1-D cubic B-Splines [5]:

$$T_{local}(x,y,z) = \sum_{i=0}^{3}\sum_{j=0}^{3}\sum_{k=0}^{3} B_i(u)B_j(v)B_k(w)\theta_{i+l,j+m,k+n} \tag{2}$$

where θ denotes the mesh of control points $n_x \times n_y \times n_z$ and

$$i = \lfloor\frac{x}{n_x}\rfloor - 1, j = \lfloor\frac{y}{n_y}\rfloor - 1, k = \lfloor\frac{z}{n_z}\rfloor - 1$$

$$u = \frac{x}{n_x} - \lfloor\frac{x}{n_x}\rfloor, v = \frac{y}{n_y} - \lfloor\frac{y}{n_y}\rfloor, w = \frac{z}{n_z} - \lfloor\frac{z}{n_z}\rfloor$$

B_i represents i^{th} basis function as follows:

$$\begin{aligned} B_0(u) &= \frac{(1-u)^3}{6}, \\ B_1(u) &= \frac{(3u)^3 - 6u^2 + 4}{6}, \\ B_2(u) &= \frac{-3u^3 + 3u^2 + 3u + 1}{6}, \\ B_3(u) &= \frac{u^3}{6}. \end{aligned} \tag{3}$$

where $u \in [0,1]$. The image intensities might change between the pre-contrast and the post-contrast image so direct image intensity comparisons as sum of squared differences (SSD) or correlation cannot be used. Normalized Mutual Information (NMI) has been recommended to be used for image alignment to avoid any dependence on the quantity of image overlap [5].

$$C_{similarity}(A,B) = \frac{H(A) + H(B)}{H(A,B)} \tag{4}$$

$$C_{smooth} = \frac{1}{V}\int_0^X \int_0^Y \int_0^Z [(\frac{\partial^2 T}{\partial x^2})^2 + (\frac{\partial^2 T}{\partial y^2})^2 + (\frac{\partial^2 T}{\partial z^2})^2 + \tag{5}$$

$$2(\frac{\partial^2 T}{\partial xy})^2 + 2(\frac{\partial^2 T}{\partial xz})^2 + 2(\frac{\partial^2 T}{\partial yz})^2]d_x d_y d_z$$

where $H(A)$ and $H(B)$ denote the marginal entropies of A and B, and $H(A, B)$ denotes their joint entropy.

It has been proposed [5] that in order to find the optimal transformation we have to minimize the cost function associated with local and global transformation parameters. The term ($C_{similarity}$), given in Equation (4), corresponds to the image similarity, while the term (C_{smooth}), given in Equation (5), corresponds to the image smoothness [5].

$$C(\theta, \phi) = -C_{similarity}(I(t_0), T(I(t))) + \lambda C_{smooth}(T) \tag{6}$$

In the above equation the weighing parameter λ shows the tradeoff between transformation smoothness and the alignment between image volumes, θ represents the global transformation, and, ϕ represents the local transformation.

3. B-Spline FFD Algorithm Implementation

We propose to implement the FFD on the PNoC using a pipelined architecture, which was initially proposed in [5] and is illustrated in Figure 3. The first stage of this pipeline, i.e., Stage 1 processes input data in fixed point format. The second to fourth stages use three pipelined multipliers and a pipelined adder to produce the end result. Stages 2 and 3 require six clock cycles each for one execution, while Stage 4 uses three execution cycles. The number of execution cycles is $N^d \times (i+1)^d \times 3$, where N is the number of pixels in the axis of the image, i denotes the B-spline order and the dimension is denoted by d. The values of basis function of splines are stored in Look Up Tables (LUTs) with 1024 entries.

The integer part of the input data points to the grid of control point stored in the external memory while fraction part directs the pointer to LUT for B-Spline. MULT_P shows the pipelined multiplier and ACC_P shows the pipelined adder. The control points (CP) from $\phi(0, 0)$ to $\theta(3, 3)$ manipulate the central 16 grey points (gp), as illustrated in Figure 4. Seven memory banks are needed for the processing of 2-D images, two for input data, two banks for the control points, one for B-Spline data, and, two for the output data.

Our Verilog code of the FFD mainly consists of three modules, i.e., B-Spline Look-Up Table (LUT), B-Spline Calculation, and the Loop Code, as shown in Fig. 5. The B-Spline LUT stores the values of the B-Spline basis functions in the memory. This way, instead of calculating the values of the B-Spline basis functions for each and every image pixel, we can directly access the value of the basis function from the LUT and thus save a considerable amount of runtime hardware resources. There are four B-Spline LUTs in our implementation that can be accessed simultaneously. The B-Spline Calculation module gets the values of B-Splines from the LUTs in the memory and assigns the respective B's on the basis of the input values of the image pixels. Basically, the module, which gets the data, reads the image file (gray scale), saved as text file in the memory, and assigns the respective values to the control

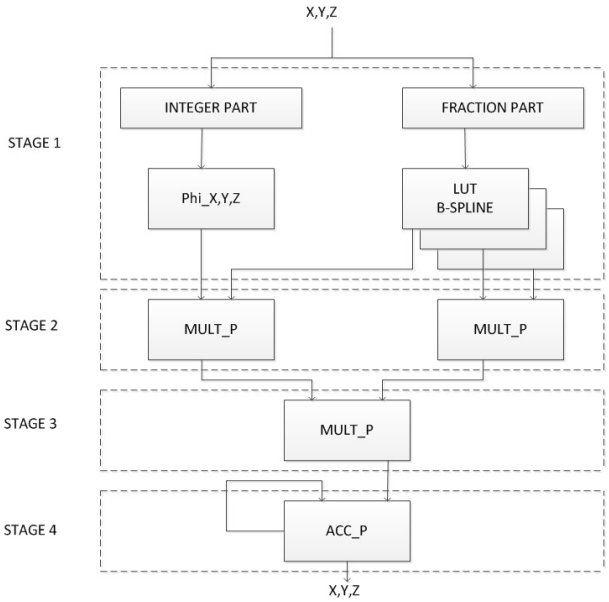

Fig. 3. Pipelined system for FFD computation [5]

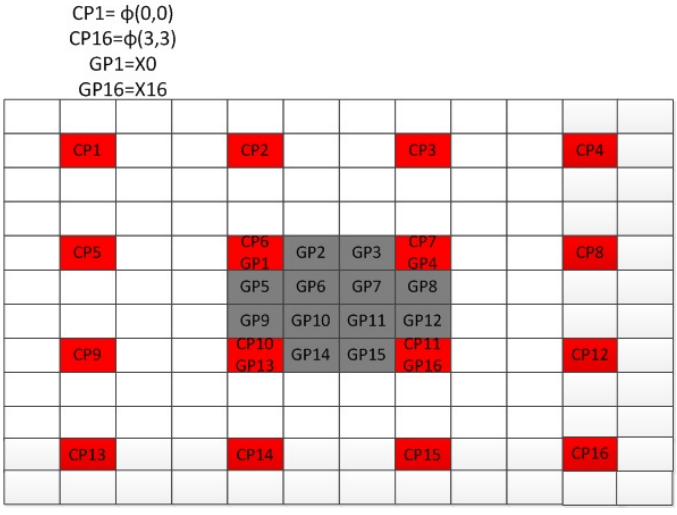

Fig. 4. Pixels of the Image under the Influence of Control Points Grid

points and the data points. Finally, the Loop Code module in Verilog calculates the effect of CPs on the image pixels based on the B-Spline values.

A simulation diagram for our FFD implementation is given in Fig. 6, with all

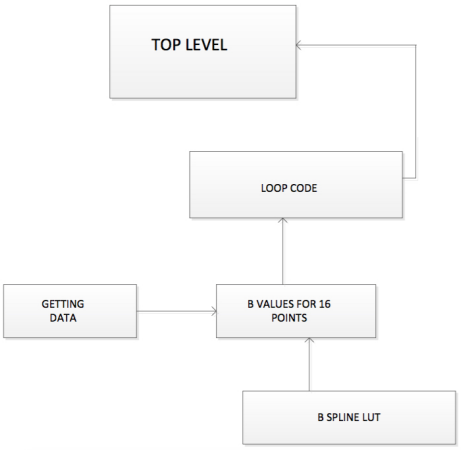

Fig. 5. Verilog Modules for the FFD Algorithm Implementation

the signals sensitive to the positive edges of the *clk* and *rst* signals. The signals $X0-X16$ and $Y0-Y16$ depict the 16 grey points of the image that have undergone the whole process of FFD. The input data is taken directly from a text file, which contains the pixels of an image, in the memory.

The basic architecture of our implementation of the PNoC for the B-Spline based architecture, as depicted in Fig. 7, consists of 16 control points for each grid of image pixels. The central 8 pixels are fed into block 1A and the effect of control point (CP1) is calculated on these pixels. The other 8 central pixels are fed into the block 1B and the effect of CP1 is calculated. The first 8 central pixels from the block 1A are fed into block 2A and the effect of CP2 is calculated on them. Similarly, pixels from 1B are fed into 2B for the calculation of effect of CP2. After 4 cycles, the effect of all the 16 control points on these pixels is calculated. All these steps are repeated until all the pixels in the image are processed. The coordination of the image data transfer between different nodes is the main design challenge of the system. There are two major communications. The first one involves the CPU to communicate with the block processors and the second one is between different block modules.

PNoC is very well suitable for this system as more than one connections are active at any instant so inter-block data transfer can occur simultaneously. The choice of window to be chosen is made by the router and no extra hardware is required for that, and, if no module is available the connection request can be queued up until the availability of a block. As this system is quite flexible so additional blocks can also be added to the present design as well.

Fig. 6. Simulation diagram for our implementation of the FFD

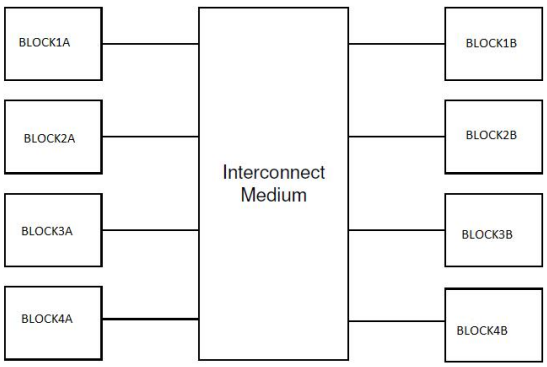

Fig. 7. PNoC architecture for FFD

4. Experimental Results

In order to illustrate the effectiveness of the proposed NoC based implementation and have a fair comparison, we developed the Verilog model of Jiang et. al's im-

Table 1. FFD implementation for 2D images (256x256 pixels)

Processor	Data Format	Clock (MHz)	Execution Time (ms)	Throughput Pixels/sec
Proposed (XC2v6000)	Fixed Point	67	60	3040400
Two Pipelines (XC2v6000)[5]	Floating Point	67	23	2791667
XC2v6000 [5]	Fixed Point	85	37	1772917
XC2v6000 [5]	Floating Point	76.3	41	1583333
Xeon (Dual) [5]	Floating Point	2666	50	1310720
AMD Athlon [5]	Floating Point	1400	100	655360
Pentium 4 [5]	Floating Point	1800	110	595782
Pentium 3 [5]	Floating Point	933	140	468114

plementation [5] of the B-Spline FFD Algorithm and synthesized it for the same FPGA target device. The data width, clock speed and area in terms of slices of our model were found to be almost the same as the ones reported previously [5]. Next, we implemented our final design for the FFD on PNoC using an XC2v6000 device and recorded its expectation time and throughput. Table 1 summarizes the results and compares the proposed PNoC based implementation of FFD with its other hardware implementations that have been reported previously [5]. It can be clearly seen that the proposed design has the highest throughput, reported in terms of pixels/sec processing, compared to all the existing FFD implementations for a 2D image. The throughput of the proposed implementation is even better than the mainstream microprocessors, which clearly indicates the potential of the proposed approach and the usefulness that it can bring in the area of medical imaging. It is worth mentioning that the rows 2-4 of Table 1 provide the figures for the implementation of the same algorithm [5] that we have implemented (Figure 3). However, our performance is better than all of them including the two-pipeline architecture. Jiang et al also implemented an alternate 2-channel architecture for processing 2-D images and achieved a performance of 4187500 pixels/sec [5]. However, this is a different design and thus its performance cannot be directly compared with the performance achieved in implementing the architecture shown in Figure 3. It may be an interesting future direction to utilize NoC for this alternate architecture and investigate the performance increase. The device utilization report, given in Table 2, indicates that the overall area is increased as compared to the Reconfigurable FPGA based implementation of Jiang [5]. This is one of the inherent drawbacks of NoC based implementations due to the additional logic that supports network communication. But, given the availability of the sub-micron transistor sizes, such an area increase is not a major concern. The increase in area makes the clock speed a

Table 2. Device (2v6000bf957-6) Utilization of FFD on PNoC

Number of Slices:	24144 out of	33792	71%
Number of Slice Flip Flops:	26588 out of	67584	40%
Number of 4 input LUTs:	5776 out of	67584	9%
Number of bonded IOBs:	393 out of	684	57%
Number of GCLKs:	1 out of	16	6%

bit slow as compared to other implementations but due to parallel processing the throughput is still higher.

5. Conclusions

The paper describes a NoC based implementation of the B-Spline FFD algorithm. For this purpose, we used PNoC, which is a very flexible architecture that suits the FPGA-based systems. Our design description is captured in Verilog language and implemented on Xilinx XC2v6000 device at 37 MHz. The proposed design is parameterizable at the compile time for the computational precisions and supports a wide range of image resolutions. The experimental results have shown a significant improvement in performance when compared with the other hardware implementations of the B-Spline FFD Algorithm.

To the best of our knowledge, this is the first implementation of a medical registration algorithm that is based on NoC. Our promising results illustrate the usefulness of NoC in the medical registration algorithms and thus other algorithms, such as non-rigid registration for breast MRI images and PETCT image registration in the chest, can also be implemented using our generic PNoC model. Another interesting future extension is to partition the images into 3 by 3 sub-images and then use 9 pipelined processors for a single block in parallel for each one of the sub-image that contains 12-bit fixed-point data. The slice resources for this system would be around 31410 for each block and it would allow us to further enhance the performance. Another interesting area of future work could be to experiment by implementing the Torus architecture instead of the 2-D mesh architecture of NoC chosen for this work. Both architectures have their own advantages as has been mentioned in [18]. The main motivation for the selection of 2-D mesh for our work was its rather straightforward implementation and better scalability in terms of area and power consumption compared to the Torus architecture. But the performance is expected to further increase if the Torus architecture is used.

References

1. H. Vesselle, D. R. Haynor, D. Mattes, T. K. Lewellen, and W. Eubank, “PETCT image registration in the chest using free-form deformations”, IEEE Transactions on Medical Imaging, volume22, pp. 120-128, 2003.

2. T. Netsch, V. Pekar, M. R. Kaus, S. Kabus, T. McNutt, and B. Fischer, "Estimation of organ motion from 4D CT for 4D radiation therapy planning of lung cancer", in Lecture Notes in Computer Science, vol. 3217: Springer-Verlag GmbH, 2004, pp. 1017-1024.
3. Rueckert D, Aljabar P, Heckemann RA, Hajnal JV, Hammers A. "Diffeomorphic Registration using B-Splines", MICCAI.;4191:702-709, 2006.
4. D. Rueckert,* L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, and D. J. Hawkes "Nonrigid registration using free-Form deformations: application to breast MR images", IEEE Transactions on Medical Imaging, volume18, pp. 8, 2009.
5. J. Jiang, W. Luk and D. Rueckert, "FPGA-based computation of free-form deformations in medical image registration", IEEE International Conference on Field-Programmable Technology, pp. 234-241, 2003.
6. W. Luk ,J. Jiang, and D. Rueckert, "An FPGA-based Computation of Free-Form Deformations", In Field-Programmable Logic and Applications, Lecture Notes in Computer Science, pages 1057 - 1061, 2003.
7. M. Haghi, A. Ghasemi, S. Moradi, "An Investigation on the Effects of Subnet Extension on Delay and Throughput in Network-on-Chip", Journal of Circuits, Systems and Computers, 25(2):1650015, 2016.
8. Md. R. Awal , M. M. H.Rahman, R. M. Nor, T. M. B. T. Sembok, M. A. H. Akhand, Architecture and Network-on-Chip Implementation of a New Hierarchical Interconnection Network, Journal of Circuits, Systems and Computers 24(2)1540006, 2015.
9. L. Benini and G. De Micheli, "Networks on Chip: a New SoC Paradigm", IEEE Computer, volume1, pp. 70-78, 2002.
10. C.Hilton and B. Nelson, "PNoC: A flexible circuit-switched NoC for FPGA-based systems", IEEE proceedings computers and digital techniques, volume153 Issue 3, 2006.
11. V. Lahtinen, E. Salminen, T. Hamalainen, and K. Kuusilinna, "Overview of bus-based system-on-chip interconnections", in Proceedings of the IEEE International Symposium on Circuits and Systems. ISCAS 02, , pp. 372-375 vol.2, 2002.
12. Hutchings, B., Bellows, P., Hawkins, J., Hemmert, S., Nelson, B., and Rytting, M., "A CAD suite for high-performance FPGA design", in Pocek, K.L., and Arnold, J.M. (Eds.). Proc. IEEE Workshop on FPGAs for Custom Computing Machines, Napa, CA, USA, pp. 175-184, 1999, (IEEE Computer Society).
13. A. Ivanov, C. Grecu, M. Jones, P. Pratim Pande, and R. Saleh, "Performance evaluation and design trade-offs for network-on-chip interconnect architectures", IEEE Transactions on Computers, volume54, no. 8, pp. 1025-1040, 2005.
14. A. Greiner, H. Charlery, A. Adriahantenaina, L. Mortiez, and C.A. Zeferino, "SPIN: a scalable, packet switched,on-chip micro-network", Proc. IEEE Conference on Design, Automation and Test, pp. 70-73, 2003.
15. B. Towles and J. W. Dally, "Route packets, not wires: On-Chip interconnection networks", Proc. IEEE International Conference on Design and Automation, pp. 684-689,2001.
16. G. K. Rauwerda, G. J. M. Smit, P. T. Wolkotte, and L. T. Smit, "An energy-efficient reconfigurable circuitswitched network-on-chip", Proc. 19th IEEE International Conference on Parallel and Distributed Processing Symposium, pp. 155-163, 2005.
17. U. Mushtaq, O. Hasan, F. Awwad,"PNOC: Implementation on Verilog for FPGA, 9th International Conference on Innovations in Information Technology, IEEE, 2013.
18. V. Sanju, N. Chiplunkar, M. Khalid, S. Joshi and J. S. Nirmala, "A Performance Study of 2D Mesh and Torus for Network on Chip Based System", Proc. International Conference on Emerging Research in Computing, Information, Communication and Applications, Elsevier, pp. 47-51, 2013.