Controlled hardware architecture for fractal image compression

Hasanujjaman*

Department of Electronics and Communication Engineering,
Kalyani Government Engineering College,
Kalyani-741235, India
Email: hasanujjaman.diet@gmail.com
*Corresponding author

Utpal Biswas

Department of Computer Science and Engineering,
University of Kalyani,
Kalyani, West Bengal, India
Email: utpal01in@yahoo.com

M.K. Naskar

Department of Electronics and Tele-Communication Engineering,
Jadavpur University,
Jadavpur, West Bengal, India
Email: mrinaletce@gmail.com

Abstract: Fractal image compression utilising algorithms have a high demand on the memory interface and the processor's arithmetic unit, which in turn fails to utilise the full capabilities of a general purpose processor. Since the algorithm is repetitive, the parallelisation reduces the time complexity of the otherwise expensive coding scheme. The design for FIC is proposed in this paper. It is based on the fact that the algorithm requires only integer arithmetic with repetitive use of the same data set. Making use of multiple functional units, controlled parallelism is introduced in this process. This makes encoding time 80 times faster than high level software implementation. It is 25 times faster than the assembly level implementation on a DSP processor.

Keywords: FIC; hardware architecture; iterated function systems; Verilog HDL.

Reference to this paper should be made as follows: Hasanujjaman, Biswas, U. and Naskar, M.K. (2020) 'Controlled hardware architecture for fractal image compression', *Int. J. Nano and Biomaterials*, Vol. 9, Nos. 1/2, pp.50–63.

Biographical notes: Hasanujjaman is currently working as an Assistant Professor in the Department of ECE in Kalyani Govt. Engineering College. He has 13 years of teaching experience. He received his BTech, and MTech from West Bengal University of Technology. He is working in the area of hardware architecture and digital system design.

Utpal Biswas received his BE, ME and PhD degrees in Computer Science and Engineering from Jadavpur University, India in 1993, 2001 and 2008 respectively. He served as a faculty member in NIT, Durgapur, India in the department of Computer Science and Engineering from 1994 to 2001. Currently, he is working as an Associate Professor in the Department of Computer Science and Engineering, University of Kalyani, West Bengal, India. He is the co-author of about 65 research articles in different journals, book chapters and conferences. His research interests include optical communication, ad-hoc and mobile communication, semantic web services, e-governance, etc.

M.K. Naskar received his BTech (Hons.) and MTech from E&ECE Department., IIT Kharagpur in 1987 and 1989 respectively and PhD from Jadavpur University in 2006. He served as a faculty member in NIT, Jamshedpur (then RIT Jamshedpur) and NIT, Durgapur (REC Durgapur) from 1991–1996 and 1996–1999 respectively. Currently, he is a Professor in the Department of Electronics and Telecommunications Engineering, Jadavpur University, Kolkata, India and in-charge of the 'Advanced Digital and Embedded Systems Lab'. His research interests include wireless sensor networks, optical networks and embedded systems.

This paper is a revised and expanded version of a paper entitled 'Hardware architecture of a decoder for fractal image compression' presented at 2019 Devices for Integrated Circuit (DevIC), Kalyani, 23–24 March 2019.

1 Introduction

The conceptual geometry of fractal image processing is mainly composed of objects such as the coastlines, mountains, boundaries, etc., wherever the application of Euclidean geometry is suitably applied. Although within the features of deterministic objectives of the fractal concept is highly comprises with a very high visual complexity whereas the content information is allocated within a very less amount of space. Other than this the concept of the self similarities and fractional dimensions are also included in the main content. However, in the concept of fractal image (Anson, 1993; Barnsley and Lyman, 1992) possess of highly redundancy in subjects of transformed copies are mostly intruded with themselves or certain parts of the major contaminants. Additionally, the concept of the fractal image compression is mainly based on the iterated function systems (IFS) (Jacquin, 1992; Barnsley and Hurd, 1992; Fisher, 1994), where the containments of the real worlds are mostly reduced upon its major contest or self- similarities presences. Moreover, the collective processes of IFS is mostly comprises with the affine transform applications where the functions of the scaling, rotation or translation of the contents are mostly initiated in the major processes.

$$w\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \boldsymbol{r}\cos\partial & -\boldsymbol{s}\sin\phi \\ \boldsymbol{r}\sin\partial & \boldsymbol{s}\cos\phi \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}$$

whereas s and r is describing the scaling factors for y and x, and the angles are for the rotation in y and x direction and the f, e are the translations on y and x, respectively.

The method of the fractal image compression is basically initiated the inverse problems for the iteration fictional theory, where the relay is assumpted with the

self-transformational exploitations based upon the block-wise redundancy basis. That further consists with transformation of contractive image in order to encode it. With the in-depth analysis for fractal compression method the demand for memory interface and processor's integer arithmetic unit is fulfilled and speedup the system (Polvere and Nappi, 2000; Saupe, 1995; Ramirez et al., 2003). Furthermore, the processor with the huge data handling capabilities is mostly complicated towards the functional units along with the biasing process towards the operations on the floating points. However, the algorithm for the fractal image compression is initiated its major applications or operations on squaring, accumulating, subtracting and shifting for the rotation and scaling of blocks. However, the major default of these processes is mostly aimed towards utilisation for the processors full capabilities (Acken et al., 1998; Panigrahy et al., 2015; Jackson et al., 2007; Vidya et al., 2000), in which the tending towards the loading factors upon simple functionality units, such as the adder, shifter, integer etc., are leadings towards resulting the inefficient usage of processor.

In the meantime it is worthwhile towards exploiting the possibilities for the specific hardware designs (Wakatani, 2012; Erra, 2005; Samavi et al., 2010) for bypassing the major drawbacks of this FIC. Additionally the implemented hardware designs that is suitably proposed the efficient work is also proposed here in this paper. That further comprises with the integral units of the arithmetic blocks in which the concepts of adder, accumulator and squarer is intruded. However, the two major aspects is also being considered here, once is towards reduction of the memory access unit and another is exploiting the concept of parallelism. Although, this concept is already proposed by another researcher, but in this particular paper the further advancements towards the decoding sections in order to reduce the drawback upon time consumptions is primarily concerned. Both of this considerations is also prompting towards the facts that is transformational over the blocks for applying repetitively upon the single range for comparing the varied number of transformable domains.

The organisation of this paper is mostly comprises with five sections, where in Section 2 the in details theoretical aspects of inverse problems based upon the iterated transformations is allocated in which the encoding and decoding is also suitable placed. In Section 3 the issues regarding the implementations such as the decompositions of non-overlapping and overlapping domain ranges are in detailed described. In Section 4, the proposed architecture related with the data flow in image processing is explained. And in the last section the presentation of the results along with the comparison is sequentially implemented.

2 Fractal image compression

The problem associated with the fractal compression method is associated with the construction of codes in any discrete original image. However, this code comprises with the lower complexity on comparison with the contents of the original image. In addition with the main contents, all of this real world images have the redundancy factors related with the affine transformations. Additionally, the basis of the transformation process is suitable because it took larger bits of real image and transforms it into smaller bits by not hampering the actual contains of the image.

Inversion problem related with the iteration transformation theory

Let (*M*, *d*) is an space metric for a digital image, whereas d describes the distortion measure. Additionally, let μ_{org} be the investigated image. The problem related with the theory can be constructed through the contractive image transformation, i.e., τ in which all of the defined mathematical forms is interrelated in an appropriate fixed point. Basically, the requirement for the transformations is defined as:

$$\forall \mu, v \in M \ \exists s < 1 : d\left(\tau(\mu), \tau(v)\right) \ll sd(\mu, v) \quad \text{and} \quad d\left(\mu_{orig}, \tau\left(\mu_{orig}\right)\right)$$

where the symbol *s* defines the contractively or the distortion measurement, this transformations is described in general as lossy, and content the lower complexity than the original image μ_{org}.

The image that is encoded further partitioned into D portions, which is determined as the domain blocks of range B*B. Another range blocks of size B/2*B/2 is also partitioned from the image that consolidated the images in actual content. However, the internal process is constructed upon findings for the match of domain and range pairs.

Disserted contractions for the affine transformations

This particular class for the discrete contractions for the affine transformations is also defined as the block-wise portion, which is the geometric and massic.

Geometric part

This part is particularly consists of spatial contractions that is operated upon the maps for the domain cells and range cells with the dimensions described above.

Massic part

This partition is used for the contracted domain in order to find the match for a given range blocks. Here, different methods for the matching purposes are being attenuated in order to shuffles the pixels for finding the range blocks.

1 Absorption

$$\theta\left(\mu_{i,j}\right) = g_0.$$

2 Luminance shift

$$\tau\left(\mu_{i,j}\right) = \mu_{i,j} + D_g$$

3 Contrast scaling

$$\sigma\left(\mu_{i,j}\right) = \alpha\mu_{i,j}$$

4 Isometries.

This particular method shuffles the image pixels into a deterministic manner that further described into eight different conical methods.

1 Identity:

$$D(i, j) = D(i, j)$$

2 Reflection about mid vertical axis:

$$D(i, nd - j + 1) = D(i, j) \quad \text{where } nd \text{ is the pixel dimension of domain}$$

3 Reflection about mid horizontal axis:

$$D(nd - i + 1, j) = D(i, j)$$

4 Reflection about first diagonal:

$$D(j, i) = D(i, j)$$

5 Reflection about second diagonal:

$$D(nd - j + 1, nd - i + 1) = D(i, j)$$

6 90° rotation:

$$D(j, nd - i + 1) = D(i, j)$$

7 180° rotation:

$$D(nd - i + 1, nd - j + 1) = D(i, j)$$

8 270° rotation:

$$D(nd - j + 1, i) = D(i, j).$$

From Table 1, it is clear that for each domain block the geometrical transformations is attenuated towards the contractions upon the domain range towards the allocated size of range blocks. Furthermore, with the massic transformations the transformed domain is obtained from the contracted domain, where the τ_i is demonstrated as the concatenated for both transformations. However, this process of matching is continued for each and every domain blocks, where the matches are associated upon the particular range blocks for the further transformations process.

In addition to the main contents it is also being intercepted that the distances between he range and domain blocks are also being computed simultaneously. In which the minimum distances between the matching partners are mostly coded as the outcomes for the transformation. This overall process is repeatedly allocated upon the range blocks, so that a less number of range blocks are entitled with the image. The equation related with this particular transformation is demonstrated as

$$\lceil \mu \rceil D_i \rceil \xrightarrow{S_i} S_i \lceil \mu \rceil D_i \rceil \xrightarrow{T_i} (T_i \circ S_i) \lceil \mu \rceil D_i \rceil = \tau_i \lceil \mu \rceil D_i \rceil = \lceil \mu \rceil R_i \rceil$$

$$\sum_{0 \leqslant i \leqslant N-1} \lceil \mu \rceil R_i \rceil = \mu.$$

Table 1 The matrices for transformation of eight symmetries

Symmetry	*Matrix*		*Description*
0	1	0	Identify
	0	1	
1	–1	0	Reflections in Y-axis
	0	1	
2	1	0	Reflection in X-axis
	0	–1	
3	0	1	Reflection about first diagonal
	1	0	
4	0	–1	Reflection about second diagonal
	–1	0	
5	0	1	
			90 degree rotation
	–1	0	
6	0	–1	
			108 degree rotation
	1	0	
7	0	–1	
			–90 degree rotation
	–1	0	

Encoding procedure

The pool for domain blocks are mostly consists with all the domain blocks in which the sets of eight different isometric transformation are also attached. Additionally, with the range blocks a pair of domain and isometric blocks are also attenuated in such away, that when the domain blocks were transformed again using the isometric method, would reveals the number of range blocks attached under the considered applications. furthermore, the mean differences attached with the average pixels values are also computed accurately in which the contracted domain blocks and the range blocks are submerged or coded all together. However, all of this data's are stored as the offset values, that also take cares for luminance, contrast scaling and absorptions transformations that are also present in between the domain and range blocks. In other words, this values under offset variables is also used in order to presence the average level of gray the image contents into its blocks and also the geometrical transformational for isometric transformations is also collared with he main contents. Therefore, the original outcomes of the process are mostly composed with the domain block, the isometric transformations and the offset values for the range blocks.

Decoding procedures

For the reconstructed process an iterative process is performed at first where the initial image with the subsequently iterations methods are submerged or convergences from the

main data. However, for the reconstructions of the range blocks, the codes are firstly examined in which the range blocks are mainly considered. In addition with the main contents the positions for the domain images are also consisted within the main contents of the data, where the offset values and the isometric values are also present within the main objectivise of the outputted data. Thus the corresponding domain blocks are also being contracted within the scaling factors that are applied for the isometric purposes in order to apply for the contacted domain blocks. Additionally the offset values are also attenuated in the main prospective which reveals the pixel values. Thus, the outcome values would be deterministic allocated for the range blocks identifications purposes, that further helps in processing the inverse process for reconstructions of the image form the range and domain blocks. Furthermore, the subsequent iterations process will be converge in order to process the inverse iteration transformations process. However, the degree attached with the closeness for decoded images is mostly depended with the accurately choosing of domain blocks and transformations of range and matches regions during the encoding procedures.

3 Implementations issue

The selectivity for shape and sizes of image pixels are mostly dependent upon several factors. However within the small image blocks

- are quite easy in order to classify the objects geometrically
- also allows a faster evaluation process for measuring the inter-block distances
- in order to encode the data, it evolves quite easy procedures with high accuracy contaminants
- that further lead a robust encoding system, that perform stability with source images along with diverse
- also allows a better redundancy upon exploiting the small areas of image
- that increases the compression rations for encoding process.

Here, in this section certain other implemented issues are also discussed briefly, in order to identify the focusable areas for handling more accurately.

The range blocks

Initially the image is separated into a number of non-overlapping bocks of square shaped and equal number of sizes. However, it can be allocated that the more or larger range regions would allocated the fewer number of blocks for further transformations can be modelled, and also lesser number of fractal files would also accommodated for the transformations process. The detailed improvers for compression ratio are presented in the lower table. Based upon the shape and sizes of the range blocks the further intruded of domain blocks would be persuade. That further increases the probabilities for finding the similar patterns of range blocks and also reduces the higher-range sizes for the confirmations of matching and transformations procedures. This particular reduction also

the fidelities he encoding process in order to reduces the data contaminants for the transformations process.

Table 2 Different prospective for choosing the shape and sizes of range blocks

Range block size 4 × 4		*8 × 8*	*16 × 16*
SNR dB	29.58	23.13	19.21
Compression ratio	1:4	1:16	1:64
Compression time(s)	1,059.03	879.4	609.11

The domain blocks

The collections for the overlapping domain blocks are also defined form the same mother images. The shape and sizes for the domain blocks would be larger then range blocks in order to placed the affine transformations of FIC based contractively. At the same time, the scaling factors associated with the scaling factors are accommodated in the main block as s < 1[7, 8]. Here, in this case the sizes of domain blocks is collaborated as B × B, for range blocks B/2 × B/2 and the contractively will be attenuated at 0.5. Technically, the larger size of domain pool will evolves the greater probability of extracting the matches for range block in order to transform the domain and also aimed towards providing a better fidelity. Hence, the further implementation sizes of domain pools would maximises the extended possibilities. It is also achieved form the domain blocks by overlapping every pixel. In addition to the main contents, the reduction upon search spaces would also reduces the compression time along with also helps in achieving the next domain blocks for further processed the transformations. Moreover, this particular process of matching the transformations process would also leads towards reducing of SNR for the overall process. Thus the compression time along with the SNR variations is also provided in the lower tables in order to provide the verification upon the overall operations.

Distortion measurement

The concept of square errors is used for measuring the distance between transformed domain and range in matrices form. It is also used for the computation process of SNR, as it evolves the architectural process of the overall operations. However, the absolute differences between the domain and range blocks are not accurately placed as because it does not lead to an appropriate domain for the selections of range blocks. Thus, a certain error upon the range and transformed domain is also being computed, that can be placed within the negligible order. Additionally, this overall processes is carried out for the further summing of inter pixel distances between the range R_m and transformed domain $\tau(D_n)$. However, the overall operated equation is given below.

$$d\left(\mu\rceil D_m, \tau\left(\mu\rceil D_n\right)\right) = \left[\sum_{0<i,\, j<B/2} \left[\left(\mu\rceil D_m\right)_{i,j} - \left[\tau\left(\mu\rceil D_n\right)\right]_{i,j}\right]^2\right]$$

Threshold

In order to process a heuristic search a typical threshold value will be used, that help in the process of the optimal matching for the domain pool and the range blocks. Additionally, the squared error that is discussed in the above section would be placed in less order of this threshold. Therefore, while the threshold would be increased than the threshold value, the matching will be stopped and also the domain and transformations pair would be sets in approximate order of range. This particular threshold value will be chosen in such a way that the SNR would be placed in the tolerable value, in order to consider the accuracy upon the decoding procedure. Within the stimulations process the accuracy of the threshold values is obtained within the range of 20 to 28, in which the value of average 24 is chosen for the threshold process. Also with this particular obtained value, the average time complexity is obtained to be reduced by second order.

Table 3 Choosing for the domain block sixe

Performance	*No. of domain blocks*		
	Start form pixel	*Start form alternative pixel*	*Start form third pixel*
SNR db	29.59	28.68	26.99
Compression time (software sequential)	1,059.03	879.4	609.11

4 Hardware implementation

Already the proposed algorithm for the establishment of FIC hardware subjectivism has already discussed in the previous sections. Although the major objectives for establishing the prospective of hardware mostly composed of integer arithmetic architectures in which the squarer, subtractor and accumulator plays the prime role for obtaining the objectives for hardware implementations. Additionally, the operations will be pursued through the control parallelism types of objectives, as because this parallelisation effects not only helps in reducing the overall time complexity for the operations, but also allows to accommodated the operations in much speeded way.

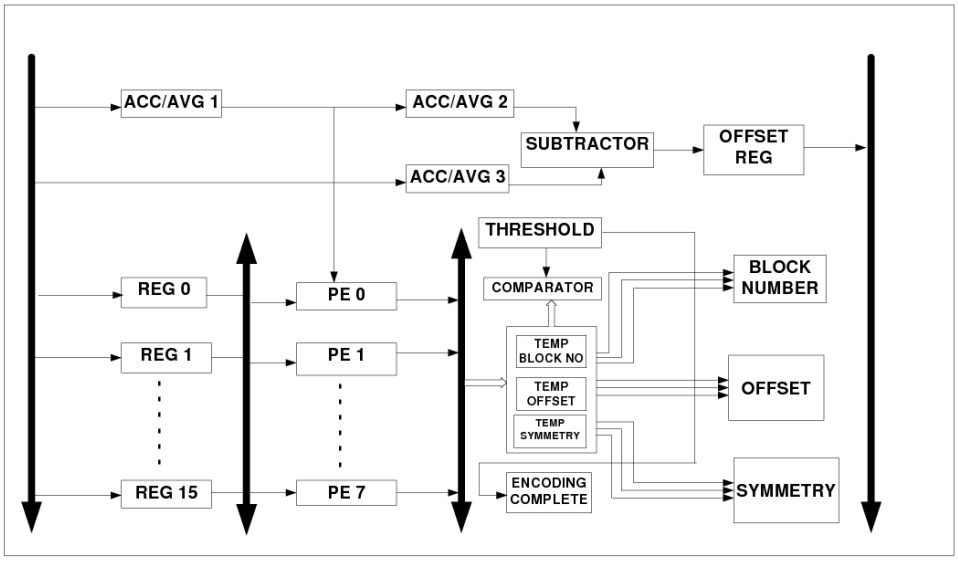
Furthermore, the memory access are also reduced enough via using the internal registers to hold large or complete sets of pixel values, that will be repeatedly used for computation of confident coefficients. With the addition of transformations block sets the reshuffling or the isomerism effects for the pixels blocks are being achieved easily. Here in this section, in-depth discussion regarding the data flow process with the encoding and decoding process is explained with the varied subsections.

The FIC architecture

In the below figures the proposed architectural blocks along with the operations processes are intercepted in detail. Sixteen registers sets are implemented for accommodated the 16 pixels values at a time for developing the range blocks.

- Eight different identical processing elements are also shown in the encoding process, that is in scripted the third figure.
- A 12-bit ACM/AVR1, accumulator/average, for operating upon the 16 domain block.
- A 10-bit ACM/AVR2, for averaging of domain blocks at 4 values of pixels.
- Another 12-bit ACM/AVR3, for finding the offsets between the range and domain blocks by accumulating and then averaging.
- A threshold register, a comparator and register.
- A control unit for coordinate the operation or generate the control signal for coders and decoder. In which the address generation logics are also included for accommodating the isometric while encoding process and also reaccepted the memory access for decoding procedures.

Figure 1 Comparator unit for range domain matching



The processing elements

There are eight processing elements available whose functionality is applying isometrics to the range to shuffle the pixels and then compare it with the contracted domain to calculate the mean square error between them. The processing element consists of a latch, subtractor, squarer and accumulator/average.

After the completion of averaging four domain pixels, the avg. output is also given to the subtractor unit which subtracts the contracted domain pixel value and a range pixel grey level. the squarer squares this subtracted value and provides input to the accumulator. After completion of 16 the cycle the accumulator provides the average of the squared error between the domain and the range.

The simple squarer design is explained below.

Architecture for the encoder

Figure 2 Hardware architecture for FIC encoder

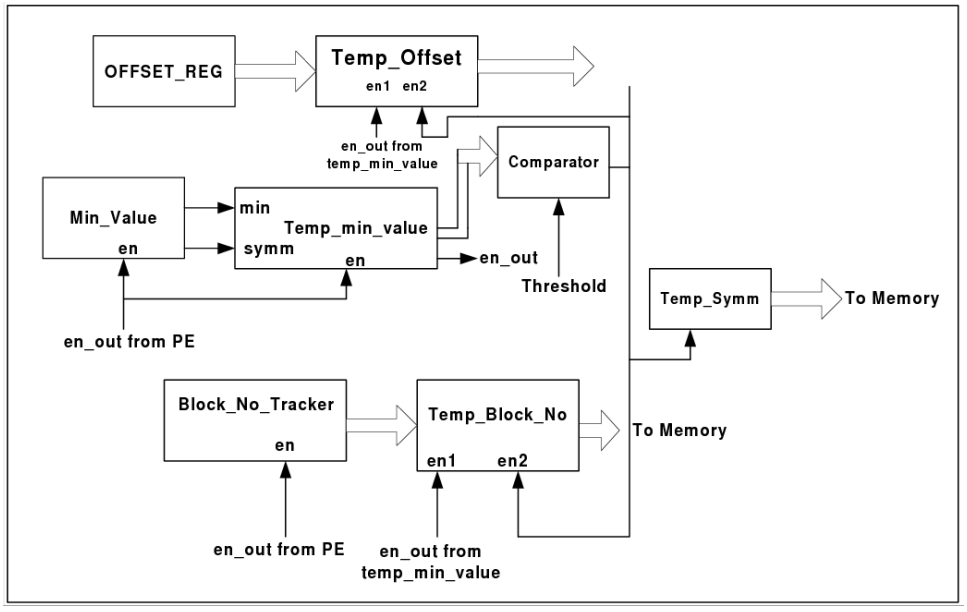
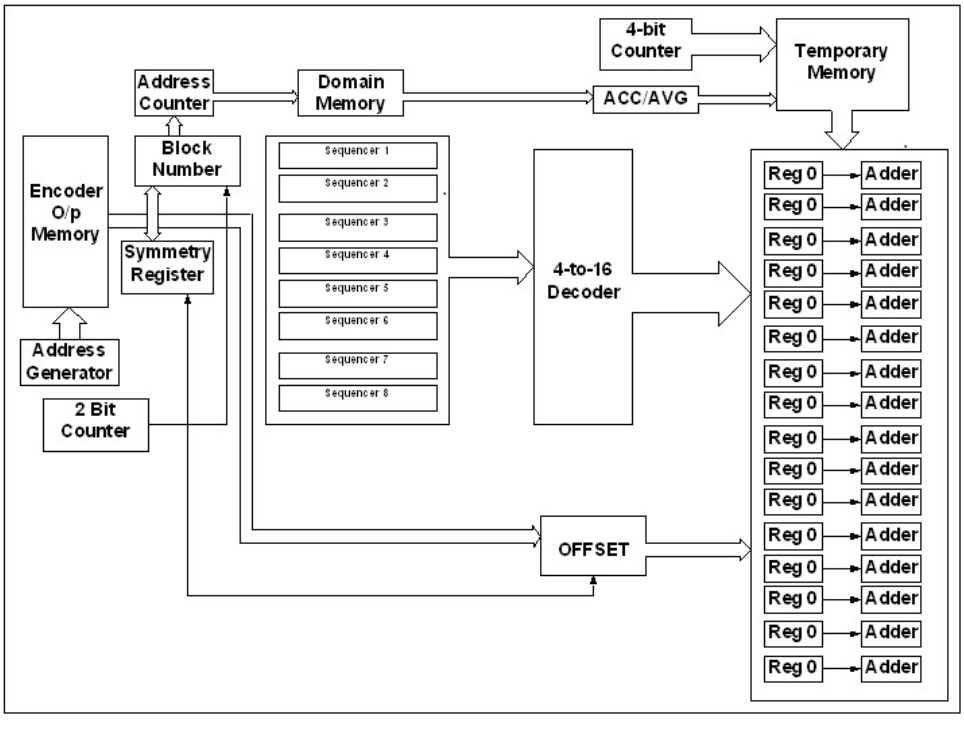


Figure 3 Hardware architecture for FIC decoder



Control unit

The control unit is a very essential part in designing the architecture of fractal image compression on FPGA, because it's that part which will control the parallelisation among several parts of the hardware. It will synchronise the operation of encoder, decoder and also different blocks in them, so we thoroughly need to assure that we don't mess it up with false timing. The term false timing is referred here because it's the clk which will decide the timing information for all the blocks in a unit. All the blocks will not get the same clk frequency instead we may need to divide the frequencies in order to provide the required timing.

Control unit may be used to provide a number of features which are listed below:

1 generate enable signals to the 16 set of registers in the encoder and decoder, so that they can be in operating condition in proper time

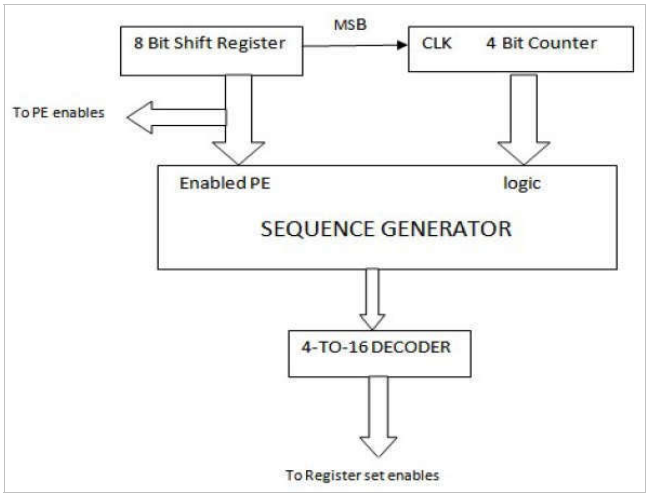
2 generate enable signals to the 8 processing elements in the encoder, so that proper isometric is applied in proper time

3 another very important fact is that we cannot keep the encoder and decoder operating at the same time, so we may also require to send proper enables to encoder and decoder too, so that we can ensure that after the completion of the encoder only we get the decoder to be in operating condition

4 after the end of encoding procedure we also require that we calculate proper domain address, the appropriate isometric to apply and also the required offset, these calculations of block no and operating symmetry is done by the control unit.

However, we have simplified the functionality available from control unit, as we have used the control unit to provide enables to the register set and processing unit of the encoder only, everything else mentioned above has been done in some different way. The architecture of the control unit is shown in Figure 4.

Figure 4 Control unit



5 Result and conclusions

The controlled hardware architecture for the FIC has been proposed above and the verilog simulated result shows that it efficiently utilised the hardware. To minimise the huge range domain matching only ten iteration is sufficient to get the result. Controlled parallelism has been incorporated to speed up the decoding process.

Figure 5 Verilog RTL schematic of FIC decoder (see online version for colours)

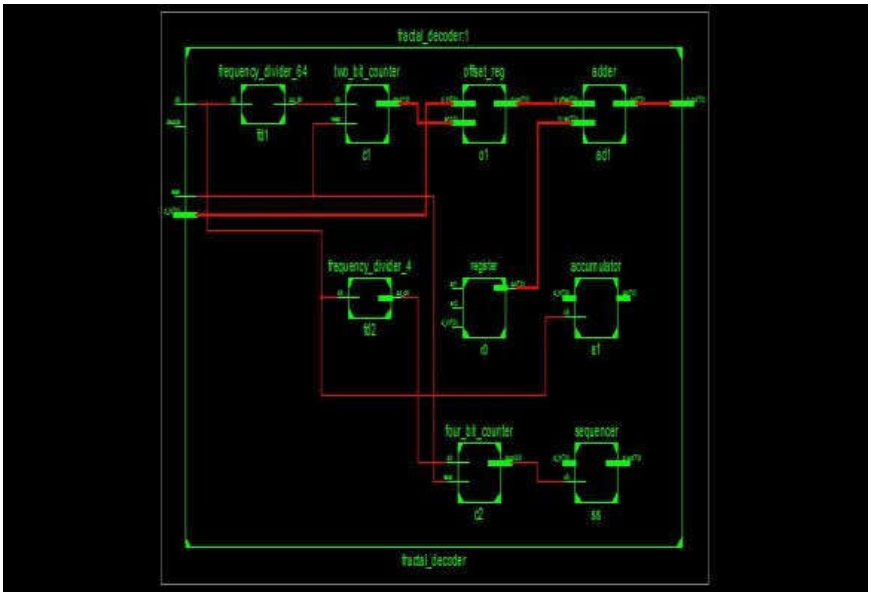


Table 4 Device utilisation summary

Slice logic utilisation	*Used*	*Available*
Number of slice registers	8	407,600
Number of slice LUTs	16	203,800
Number of occupied slices	11	50,950
Number of fully used LUT-FF pairs	8	16
Number of bonded IOBs	18	500
Number of ILOGICE2/ILOGICE3/ISERDESE2s	8	500
Average fanout of non-clock nets	1.78	

From Table 4 point of discussion and the mathematical adherences, it can be concluded that the successful implementations of the hardware's are achieved for the in general applications of fractal image compression. Additionally, from the point of exempted results, it is also being achieved that the process of bits transactions along with the hardware applications provides much better results than the other corresponding paper works. In addition to the main contents, the successful implementations of the subtraction and the comparators help the overall process to achieve a much higher rate of speeds

within the operation. Thus, the impractical applications of the fractal image compression are achieved from the overall discussions.

References

Acken, K.P., Irwin, M.J. and Owens, R.M. and A parallel ASIC architecture for efficient fractal image coding', *Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology*, Vol. 19, No. 2, pp.97–113.

Anson, L.F. (1993) 'Fractal image compression', *Journal BYTE*, October, Vol. 18, No. 11, pp.195–202, McGraw-Hill, Inc. Hightstown, NJ, USA.

Barnsley, M.F. and Hurd, L.P. (1992) *Fractal Image Compression*, A.K. Peters Ltd. Natick, Boston, MA, USA ©1993, ISBN: 1-56881-000-8.

Barnsley, M.F. and Lyman, P.H. (1992) *Fractal Image Compression*, Academic Press, New York.

Erra, U. (2005) 'Toward real time fractal image compression using graphics hardware', *International Symposium on Visual Computing*, Springer, pp.723–728, doi:10.1007/11595755_92.

Fisher, Y. (1994) *Fractal Image Compression: Theory and Application*, doi:10.1142/S0218348X94000442.

Jackson, D.J., Ren, H. Wu, X. and Ricks, K.G. (2007) 'A hardware architecture for real-time image compression using a searchless fractal image coding method', *J. Real Time Image Process*, Vol. 1, No. 3, pp.225–237, doi: 10.1007/s11554-007-0024-2

Jacquin, A.E. (1992) 'Image coding based on a fractal theory of iterated contractive image transformations', *IEEE Trans. Image Process*, Vol. 1, No. 1, pp.18–30, doi:10.1109/83.128028.

Panigrahy, M., Chakrabarti, I. and Dhar, A. (2015) 'Low-delay parallel architecture for fractal image compression', *Circuits Syst. Signal Process,* pp.1–21, doi:10.1007/s00034-015-0088-3.

Polvere, M. and Nappi, M. (200) 'Speed-up in fractal image coding: comparison of methods', *IEEE Trans. Image Process*, Vol. 9, No. 6, pp.1002–1009, doi:10.1109/83.846243.

Ramirez, A.M., Sanchez, A.D., Aranda, M.L. and Vega-Pineda, J. (2003) 'Simple and fast fractal image compression for VLSI circuits', *Proceedings of the 3rd International Symposium on Image and Signal Processing and Analysis*, pp.112–116.

Samavi, S., Habibi, M., Shirani, S. and Rowshanbin, N. (2010) 'Realtime fractal image coder based on characteristic vector matching', *Image Vision Comput.*, Vol. 28, No. 11, pp.1557–1568, doi:10.1016/j.imavis.2010.03.011.

Saupe, D. (1995) 'Accelerating fractal image compression by multi-dimensional nearest neighbour search', *Data Compression Conference, 1995. DCC'95*. Proceedings, IEEE pp.222–231, doi:10.1109/DCC.1995.515512.

Vidya, D., Parthasarathy, R., Bina, T. and Swaroopa, N. (2000) 'Architecture for fractal image compression', *J. Syst. Archit.*, Vol. 46, No. 14, pp.1275–1291, doi:10.1016/S1383-7621(00)00018-7.

Wakatani, A. (2012) 'Implementation of fractal image coding for GPGPU systems and its power-aware evaluation', *2012 IEEE International Systems Conference (SysCon)*, IEEE, pp.1–5, doi:10.1109/SysCon.2012.6189434.