A Hardware Area Efficient Implementation of Discrete Fourier Transform on FPGA for Image Processing Applications

Deepshikha Shurvansi[1], Sachin Bandewar[2]
[1,2]*Department of ECE, Sri Satya Sai College of Engineering, RKDF University, Bhopal, M.P*
[1]*96deepshikha@gmail.com*,[2]*sachin.bandewar9@gmail.com*

ABSTRACT
Very large scale integration (VLSI) implementation of digital image processing application is growing rapidly due to its great demand in consumer electronic devices. Compression is useful as it helps in reduction of the usage of expensive resources, such as memory (hard disks), or the transmission bandwidth required. But on the downside, compression techniques result in distortion (due to lossy compression schemes) and also additional computational resources are required for compression-decompression of the data. Reduction of these resources by comparing different algorithms for DFT is required. In this paper, FPGA Implementations of different algorithms for 1-DFT using VHDL has been carried out using XILINX ISE Design suite as the synthesis tool with post route verification using Virtex 7 device. The hardware consumption of the design is also presented which gives the optimum technique for compression. Finally VLSI implementation of 2-D DFT is carried out using the optimum 1-D technique for 8x8 matrix input. The results obtained are discussed and improvements are suggested to further optimize the design.

Keywords: Discrete Fourier Transform, VLSI Architectures, FPGA, VHDL, XILINX ISE Suite

I. INTRODUCTION

Image compression, the art and science of reducing the amount of data required to represent an image, is one of the most useful and commercially successful technologies in the field of digital image processing. Digital image and video compression is now very essential. Internet teleconferencing, High Definition Television (HDTV), satellite communications and digital storage of movies would not be feasible unless a high degree of compression is achieved.

Compression is useful as it helps in reduction of the usage of expensive resources, such as memory (hard disks), or the transmission bandwidth required. In today‟s age of competition where everything is reducing its size every minute, the smaller is the better. But on the downside, compression techniques result in distortion (due to lossy compression schemes) and also additional computational resources are required for compression-decompression of the data..

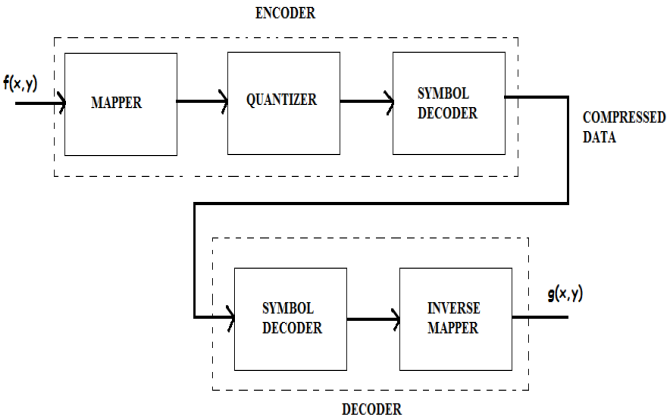

Figure 1: Functional block diagram of a general image compression system

The DFT belongs to the family of frequencytransforms that map temporal or spatial functionsinto frequency functions. .The DFT accomplishesthis in amanner similar to the better- known FourierTransform. The significant difference between theDiscrete Fourier Transform (DFT) and DFT'salternative is that the DFT usesonly real values,i.e., no complex numbers. The DFT achieves this via the kernel or ***cas***function:Baugh-Wooley algorithm for multiplication.

It is an algorithm for high-speed, two"s complement, m-bit by n-bit parallel multiplication. The two"s complement multiplication is converted to an equivalent parallel array addition problem in which each partial product is the AND of a multiplier bit and a multiplicand bit, and the signs of all the partial product bits are positive [7].

The algorithm"s principle advantage is that the signs of all the partial products are positive, allowing the product to be formed using array addition techniques. Therefore the product is formed with only the AND function and the ADD function. No subtraction is necessary, nor is the NAND function needed. For 8x8 bit multiplier, the output is a 16-bit binary number.

It computes DFT in 5 pipelined stages. For first two stages, it consists of two 4-pointDFT modules that receive the odd and even indexed subsequences and$\boldsymbol{x_2}$and from the input buffer. In the third pipelined stage, multiplication with 1/ 2 is done for the required coefficients i.e. X_{21} andX_{23} . Next they are added and subtracted in the fourth stage. During 3rd and 4th stages the rest of the coefficients are passed through a delay. Delay consists of simply registers i.e. they are stored in different registers and passed to the next stage. Finally the fifth pipelined stage is a parallel adder block which adds/subtracts the coefficients to give the desire output.

The block diagram of the described method is given in figure 1.1.

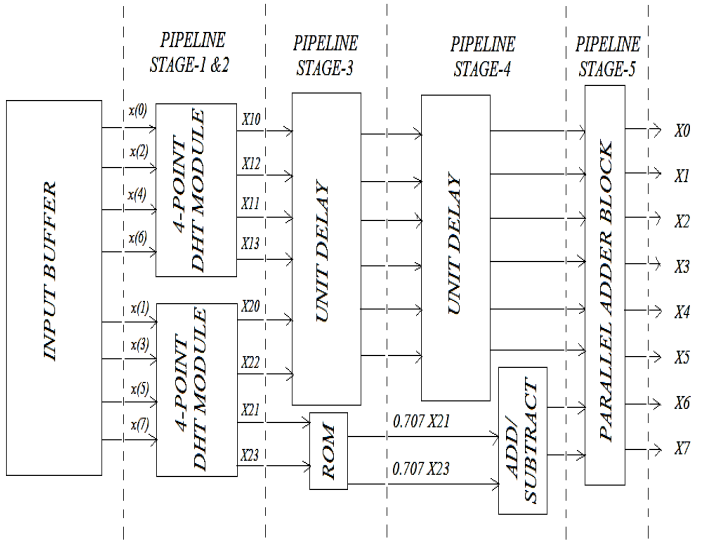

Figure 2: Flow chart of the 8-point DFT in pipelined approach with delays

II Architecture

The figure 2.1 illustrates the design flow system to implement the architecture of the 2-D DFT. The architecture has been implemented using the 1-D DFT blocks as components and various shift registers to smoothly run the entire operation. The input matrix has to be fed row-wise to the FPGA since it cannot take such a large input matrix at a time.

III Xilinx Simulation Results and Discussion

3.1 Design Summary for different Architectures

This section presents simulation results of the proposed architecture on Xilinx ISE Design suite. The results show that the architecture proposed is hardware efficient, and can be used for hardware accelations in protable electronic devices.

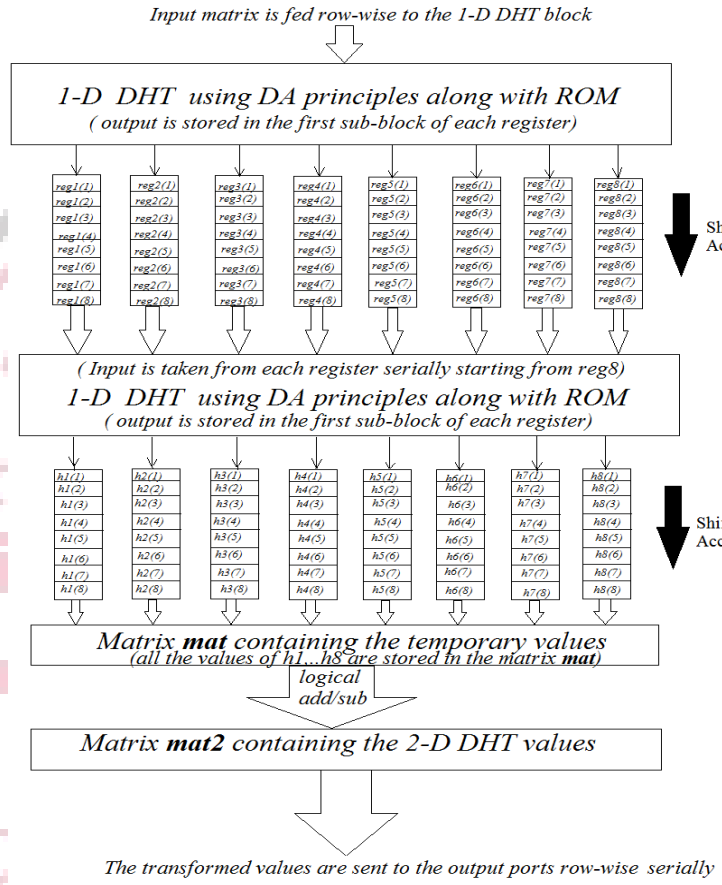

Figure 3: Flow diagram of 2-D DFT implemntation

Table 1: MSE and PSNR tabulated for Lena and Baboon images.

IMAGE	Threshold value as % of normalized energy	10%	20%	40%	60%	80%	100%
LENA	MSE	21.4292	31.7735	45.7513	55.7382	66.5523	69.6302
	PSNR	80.1777	76.2389	72.5931	70.6186	68.8454	68.3933
BABOON	MSE	67.5956	101.201	140.805	165.766	184.946	204.709
	PSNR	68.6898	64.6541	61.3515	59.7195	58.6246	57.6094

Table 2: Design Summary for DA architecture for 3-point DFT

Logic utilization	Used	Available	Utilization
Number of slices	428	4656	9%
Number of slice flip-flops	359	9312	3%
Number of 4-input LUTs	783	9312	8%
Number of bonded IOBs	73	232	31%
Number of GCLKs	1	24	4%

Table 3: Design Summary for architecture of 8-pt DFT by DA (8-bit to 10-bit)

Logic utilization	Used	Available	Utilization
Number of slices	121	4656	2%
Number of slice flip-flops	143	9312	1%
Number of 4-input LUTs	208	9312	2%
Number of bonded IOBs	77	232	33%
Number of GCLKs	1	24	4%

Table 4: Design Summary for architecture of 8-pt DFT by DA (8-bit to 10-bit)

Logic utilization	Used	Available	Utilization
Number of slices	516	13697	3%
Number of slice flip-flops	666	27392	2%
Number of 4-input LUTs	795	27392	2%
Number of bonded IOBs	195	556	35%
Number of GCLKs	1	16	6%

Table 5: Design Summary for architecture of 8-pt DFT by DA (8-bit to 10-bit)

Logic utilization	Used	Available	Utilization
Number of slices	432	13697	3%
Number of slice flip-flops	355	27392	1%
Number of 4-input LUTs	694	27392	2%
Number of bonded IOBs	179	556	32%
Number of GCLKs	1	16	6%

We can clearly see that hardware used for DA architecture is much less than SA architecture. This difference will only increase as we increase the number of inputs i.e. for 8-point DFT the silicon used in DA will be much less than that used in SA. This is due to the reason that there is no multiplication or any other higher calculations involved in DA model. It comprises only of adders and shift registers. Hence DA architectures are faster, low power and more compact than SA architectures. bit vector and Output is 10 bit vector)

Figure 4 presents Design Summary for architecture of 8-pt DFT by DA (8-bit to 10-bit)

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	4,610	27,392	16%	
Number of 4 input LUTs	5,059	27,392	18%	
Logic Distribution				
Number of occupied Slices	4,268	13,696	31%	
Number of Slices containing only related logic	4,268	4,268	100%	
Number of Slices containing unrelated logic	0	4,268	0%	
Total Number of 4 input LUTs	5,139	27,392	18%	
Number used as logic	4,727			
Number used as a route-thru	81			
Number used as Shift registers	331			
Number of bonded IOBs	97	556	17%	
Number of RAMB16s	129	136	94%	
Number of BUFGMUXs	2	16	12%	
Number of BSCANs	1	1	100%	
Number of RPM macros	14			

Figure 4: Design Summary for architecture of 8-pt DFT by DA (8-bit to 10-bit)

Table 6: Power analysis of the 2-D architecture

Power summary	I(mA)	P(mW)
Total estimated power consumption	--	103
Total Vccaux 2.50V	10	25
Total Vcco25 2.50V	1	3
Quiescent Vccint 1.50V	50	75
Quiescent Vccaux 2.50V	10	25
Quiescent Vcco25 2.50V	1	3

Figure 4 presents Design Summary for architecture of 8-pt DFT by DA (8-bit to 10-bit)The silicon utilization in the architecture with pipelined stages is higher than in the architecture using only DA principles with ROM. Also the computational time is more in pipelined stage architecture due to the delays introduced in the model of the method. Hence, architecture based on DA principles using ROM is more efficient and is used for the modeling of 2-D DFT. Another architecture which inputs 10-bits vectors and gives 12-bit vectors is also implemented, which also is more efficient than the pipelined stage architecture. 2-D DFT is calculated for 8x8 matrices inMatlab and Xilinx. The simulation results of both are shown for two different inputs. The matrix **mat** contains the 2-D DFT for the input given to **x1- x8** registers in the Xilinx simulation results shown below. For Matlab simulation, **f** contains the input matrix and **h** gives the output matrix. We can see that the results obtained from the hardware implementation are same for the odd numbered columns and a little error is generated in even-numbered columns. This is due to the reason that the even-numbered columns of the kernel matrix generated from the “cas” function has mixed fraction values which have been approximated. This approximation can be reduced if we use binary representation of decimal values for calculation purposes.

IV CONCLUSION AND FUTURE WORK

In the present work, two-dimensional Discrete Fourier Transform for an 8x8 input matrix was implemented in FPGA using VHDL as the synthesis tool. The 1-D DFT was also calculated for 8-point input using two algorithms and their effectiveness were discussed. It is shown that the DA approach provides better performance in terms of speed and area when is compared with the pipelined approach. This primarily focuses on image compression with less computation and low power. The simulation results and design summary for 2-D DFT were obtained andit was shown that the architecture implemented is an efficient method which uses limited space and time. The hardware utilization is quite optimum and power analysis shows that the power requirement is also optimum. However if the input contents are large, they tend to overflow from the registers and hence error occurs. It can be rectified by saving the transformed coefficients in larger registers. Also due to quantization in the contents of the ROM, even-number outputs are more deviated from the desired results than the odd-numbered outputs. This is due to the reason that even numbered columns of the transform kernel consist of mixed fractions which are rounded off to be store in the ROM registers. This drawback can be removed if the decimal fractions are converted to binary representation before being stored. Also, a lot of memory is used in this architecture. It can be solved by using the ROM-free DA technique. These are some of the improvements that can be done to the improvise the design.

REFRENCES

[1] Pyrgas, Labros, Paris Kitsos, and Athanassios N. Skodras. "An FPGA design for the two-band fast discrete Fourier transform." 2016 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)., Cyprus, pp295-299, 2016.

[2] Parsai, Shirali, Swapnil Jain, and Jyoti Dangi. "VHDL implementation of Discrete Fourier Transform using Urdhwa multiplier." 2015 IEEE Bombay Section Symposium (IBSS). pp. 1-6, 2015.

[3] Chiper, Doru Florin. "A novel VLSI DFT algorithm for a highly modular and parallel architecture." IEEE Transactions on Circuits and Systems II: Express Briefs 60.5 (2013): 282-286.

[4] Huddar, Sushma R., et al. "Novel high speed vedic mathematics multiplier using compressors." 2013 International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s). pp.465-469, 2013.

[5] Kerur, S. S., et al. "Implementation of Vedic multiplier for digital signal processing." International Conference on VLSI, Communication & Instrumentation (ICVCI). pp. 1-6, 2011.

[6] Jagadguru Swami Sri Bharati Krishna Tirthaji Maharaja, "Vedic Mathematics: Sixteen simple Mathematical Formulae from the Veda", Delhi, pp. 1-6 ,2011.

[7] Sumit Vaidya and Depak Dandekar. "Delay-power perfor-mance comparison of multipliers in VLSI circuit design". International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4, p.p 151-158, July 2012.

[8] SumitVaidya and DepakDandekar. "Delay-power perfor-mance comparison of multipliers in VLSI circuit design", International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4, p.p. 64-72, July 2012.

[9] Chidgupkar, Purushottam D., and Mangesh T. Karad. "The implementation of vedic algorithms in digital signal processing." Global J. of Engng. Educ8.2 (2004), Australia, p.p 153-158, 2004.

[10] S. Bouguezel, M. O. Ahmad, and M. N. S. Swamy, "New parametric discrete Fourier and Fourier transforms, and algorithms for fast computation," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 58, no. 3, pp. 562–575, Mar. 2011.

[11] J. S. Wu, H. Z. Shu, L. Senhadji, and L. M. Luo, "Radix 3 × 3 algorithm for the 2-D discrete Fourier transform," IEEE Trans. Circuits Syst. II, Exp.Briefs, vol. 55, no. 6, pp. 566–570, Jun. 2008.

[12] S. Bouguezel, M. O. Ahmad, and M. N. S. Swamy, "A split vector-radix algorithm for the 3-D discrete Fourier transform," IEEE Trans. CircuitsSyst. I, Reg. Papers, vol. 53, no. 9, pp. 1966–1976, Sep. 2006.

[13] D. F. Chiper, "Radix-2 fast algorithm for computing discrete Fourier transform of type III," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 59, no. 5, pp. 297–301, May 2012.

[14] H. Z. Shu, J. S. Wu, C. F. Yang, and L. Senhadji, "Fast radix-3 algorithm for the generalized discrete Fourier transform of type II," IEEE SignalProcess. Lett., vol. 19, no. 6, pp. 348–351, Jun. 2012.

[15] D. F. Chiper, "Fast radix-2 algorithm for the discrete Fourier transform of type II," IEEE Signal Process. Lett., vol. 18, no. 11, pp. 687–689, Nov. 2011.
[16] R.N. Bracewell, The Discrete Fourier Transform, J. Opt. Soc. Amer., vol.73, pp.1832-1835,Dec., 1983.
[17] R.V.L. Fourier, A More Symmetrical Fourier Analysis Applied to Transmission Problems, Proc. IRE,vol.30, pp.144-150, Mar., 1942.
[18] R.N. Bracewell, The Fourier Transform, Oxford University Press, 1986.
[19] K.J. Olejniczak, G.T. Heydt Eds., Proc. of the IEEE, Section on the Fourier Transform, vol.82, n.3, Mar., pp.372-447, 1994.
[20] J.-L. Wu and Shiu, Discrete Fourier Transform in Error Control Coding, IEEE Trans. Acoust., Speech, Signal Processing, ASSP-39, pp.2356-2359, Oct., 1991.
[21] R.E. Blahut, Fast Algorithms for Digital Signal Processing, Addison-Wesley, 1985.
[22] Bi, Guoan, and Yan Qiu Chen. "Fast DFT algorithms for length N= q* 2/sup m." IEEE transactions on signal processing 47.3 (1999): p.p 900-903, 1999.
[23] Ravi Singh Pippal, "A Novel Smart Card Authentication Scheme using Invisible Image Watermarking", SHODH SANGAM - A RKDF University Journal of Science and Engineering, Vol. 01, No. 01, 2018.