Programmable N-Point FFT Design

Ramesh Bhakthavatchalu, Meera Viswanath, Pallavi Venugopal M., and Anju R.

Abstract—The FFT, Fast Fourier Transform is the most ubiquitous algorithm used for signal analysis in present-day communication systems (Example: OFDM). The general principle of FFT Algorithms is to use a Divide and Conquer approach that effectively reduces operation count. The FFT is generally computed either using a single radix or by mixed radix algorithms. Several optimization techniques have been proposed to increase efficiency while computing the Fourier Transform. This paper develops a reconfigurable FFT processor that can take any number of points as input, and is not limited to sequence lengths that are powers of the radix. The paper compares the factorized mixed radix approach with the single radix approach for computation of the FFT of a truly N-point sequence and shows that the mixed-radix approach yields better performance results. The test designs were developed in VHDL, verified in Matlab and Modelsim, and implemented using Xilinx Virtex5 LX110T FPGA.

Index Terms—Fast fourier transform, mixed radix, cooley-Tukey algorithm, prime-factor algorithm, word length effects.

I. INTRODUCTION

Frequency domain analysis of signals has certain advantages over the time domain approach. The Fourier transform is a tool used to obtain the frequency domain counterparts of time domain signals. The number of computations involved in the direct calculation of the Discrete Fourier Transform increases significantly with an increase in the length of the input sequence. FFT Algorithms can compute the coefficients of the DFT at a much faster rate due to the reduced number of computations since redundant calculations are eliminated in the process [1]-[3], thus increasing the processing speed dramatically [4], [5]. The FFT carries out the computations iteratively resulting in a considerable savings in computation time using any one of several efficient algorithms employed to compute the DFT coefficients.

The method used in FFT algorithms involves a linear change of index variables to map the one dimensional problem into a multi-dimensional problem [6]. Most FFT processors are either designed for fixed lengths or are programmable, but can compute the FFT only for sequences that are of lengths that are powers of two. The design of a 3780 point FFT processor for applications like TDS-OFDM, using a mixed radix approach is discussed in [7]. Ref. [8] describes the implementation of radix-2² single-path delay feedback pipelined FFT/IFFT processor. The merits of R2²SDF FFT and the limited resource utilization are

discussed in [9]. The area of variable length FFT Processors has been largely unexplored. A variable-length FFT processor that integrates two radix-2 stages and three radix-2³ stages for FFT sizes 512, 1024 and 2048 was proposed in [10]. Ref. [11] describes the design of a reconfigurable VLSI architecture for an FFT processor, but is restricted to powers of 2. In these processors, the technique of zero padding is typically employed to first extend the sequence length to a power of 2, if it is not already so, and then compute the FFT. The disadvantage of this technique is that the number of zeros padded to the sequence increases tremendously as N increases resulting in shifts in the frequency spectrum. A sequence of length 514 would have to be zero-padded and converted to a 1024 point sequence before computing its FFT.

This paper presents a methodology to find the FFT of an 'N' point sequence where N can be any integer and not necessarily a power of two or the specific radix used. Highly optimized FFT blocks for smaller lengths (2 to 8) are designed. 'N' is decomposed into various factors and the FFTs of these smaller length factors are then computed using the basic building blocks. Since a mixed radix approach is used, zero-padding would not be required in a majority of the cases. Even in those cases where the FFT has to be computed for sequences that are of prime number lengths, much fewer zeros would need to be appended to the sequence. The next section gives an overview of the Fast Fourier Transform and Radix-R Butterfly based architecture. Section III discusses our proposed design in detail. Some synthesis results are presented in section IV.

II. BACKGROUND

A. Fast Fourier Transform (FFT)

The Discrete Fourier Transform for a sequence of length 'N' is defined as

$$x(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi nk/N} \quad (1)$$

where n , k and N are integers and $j = \sqrt{-1}$. $k=0,1,...(N-1)$. $W_N = e^{-(j2\pi/N)}$ represents the twiddle factor. If the N transform values are directly calculated from the N values of input, N² complex multiplications and N*(N-1) complex additions are required. As N increases, the number of computations increases tremendously affecting the system efficiency. One of the techniques used to improve the system performance is index mapping where a one dimensional FFT is converted to a higher-dimensional FFT. The Cooley-Tukey algorithm [12] which uses a divide and conquer approach recursively partitions a sequence of length N as $L \times M$. For $N=M \times L$ and

Manuscript received September 20, 2013; revised February 26, 2014.

The authors are with Department of Electronics and Communication Engineering Amrita Vishwa Vidyapeetham, India (e-mail: ramesh.amrita@gmail.com).

$k=Mp+q$ where $0 \leq p \leq (L-1)$ and $0 \leq q \leq (M-1)$, The FFT can now be represented as in (2). In this method, the computation of the Fourier Transform coefficients broadly involves three steps - 1) Compute L M-point DFTs. 2) Multiply the output of this stage with twiddle factors. 3) Compute M L-point DFTs. The number of multiplications is now reduced

$$X[k] = \sum_{l=0}^{L-1} \sum_{m=0}^{M-1} x[l, m] W_M^{mq} W_N^{lq} W_L^{lp} \quad (2)$$

to $N(M+L+1)$ and the number of additions is reduced to $N(M+L-2)$. The Symmetry ($W_N^{(k+N/2)} = -W_N^k$) and periodicity ($W_N^{(k+N)} = W_N^k$) properties of the twiddle factors can be used for more efficient computation of the DFT.

Mixed Radix algorithms factorize a large data vector into multiple sequences of shorter lengths. These smaller blocks can in turn be multiplied together to compute the DFT of larger length sequences. Hardware complexity can be minimized by reducing the number of multiplication involved in the computation of the Fourier Coefficients. Also, the use of a higher radix is regarded to be more efficient due to the reduced number of multiplications [13]. While the Prime-factor algorithm has the advantage that there are no intermediate twiddle factor multiplications, it can be used only in those cases when the factors of 'N' are relatively prime. Also, the Prime factor algorithm involves a more complex reordering of the sequence at the output based on an index-mapping that makes use of the Euler Totient function.

B. Radix-r Butterfly Based Architecture

Radix-2 Algorithms can be used only for lengths that are powers of two. The basic module involves addition and subtraction. In general, for a radix-R algorithm $(N/R) \log_R N$ multiplications and $N \log_R N$ additions are required. There are N/R butterflies per stage, and each butterfly requires $(R-1)$ complex multiplications. The precision of the values can be improved if the number of bits allotted is increased on the word length thereby improving the efficiency of the system. As the word length increases the complexity of the system increases thereby increasing the size of the memory and computational units and this in turn results in increased consumption of the power and area. The size of word length selection creates a delicate trade-off between precision and complexity. The word length for fixed and floating point implementations can be selected based on mixed-signal bit-true simulation of the whole system [14].

C. Multi-Dimensional Index Mapping

Multidimensional index mapping is used to uncouple the calculations of the discrete Fourier Transform. It is often easy to translate an algorithm using index mapping into an efficient program. Index maps are further classified, depending on whether the factors of 'N' are mutually prime or not. Separate index maps for the 'n' and 'k' terms in the basic DFT expression, each of which follow specific criteria, may be defined.

III. PROPOSED DESIGN

The architecture proposed for computing the FFT of any N

point sequence involves the mixed radix approach. The design mainly involves three stages- 1) Factorization of the number 'N' 2) The decision to choose between the Cooley-Tukey and Prime Factor Algorithm for FFT Computation depending on the factors of N. 3) choice of the sequential order of the factors with regard to computation of the N-point FFT. The basic building blocks used in the design are optimized for high speed and minimal area and are designed for Radices 2 through 8. The implementation of the radix 5 block is diagrammatically represented in Fig.1.

Any N point sequence is factorized as a combination of these basic radices and decomposed into various factors N_1, N_2, N_3 , etc. The FFT for the N point sequence is then calculated by computing the N point FFT for radix N_1, N_2, N_3 , etc., sequentially. These factors of N forms the basis for choosing between the Cooley-Tukey FFT Algorithm and the Good-Thomas Prime Factor FFT Algorithm. Unlike the Cooley-Tukey approach, PFA does not involve any intermediate multiplication between the two successive stages.

Fig. 1. Diagrammatic representation of Radix 5.

Fig. 2. Block diagram representation of Cooley-Tukey algorithm when $N=N_1 \times N_2 \times N_3$.

The order in which the basic building blocks are used depends on the order of the factors. The initial N point sequence is given as input to the first block (radix N_1). The output from the same is multiplied with appropriate twiddle factors (if the Cooley-Tukey approach is used) or may directly be given (if PFA) as input to the successive stage (radix N_2). The FFT coefficients of the N point sequence will be obtained once the sequence passes through all the building blocks ie; radices N_1, N_2, N_3 , etc. A reordering of the input may be required in some intermediate stages depending on the radix used in the previous stage.

Since mixed radices are implemented, the probability of having to use any zero padding is considerably reduced and

even for those sequence lengths that will require zero-padding prior to FFT computation, the number of zeros that are required to be padded in the design will be much lesser as compared to an approach involving a single radix. Also, unlike the single radix approach, the number of computations involved in a mixed radix approach is reduced and this further increases the efficiency of the proposed design.

The design can also choose between using 2 or 3 factors to decompose the larger composite number and compute its FFT. While the Prime-Factor algorithm requires no intermediate twiddle factor multiplications, the Cooley-Tukey algorithm approach when N is factorised as $N=N_1 \times N_2$ requires one intermediate twiddle factor multiplication and the Cooley-Tukey algorithm approach when N is factorized as $N=N_1 \times N_2 \times N_3$ requires three intermediate twiddle-factor multiplications. Fig. 2 shows the block diagram representing all the different stages involved in the computation of the FFT of an N -point sequence when N is factorized as $N=N_1 \times N_2 \times N_3$.

IV. SYNTHESIS RESULTS

The basic building blocks were implemented in VHDL and synthesised in Xilinx Virtex 5 LX110T family of FPGA. The Xilinx results are shown in Table II.

TABLE I: COMPARISON OF SYNTHESIS RESULTS FOR AN 8-POINT SEQUENCE

Slice Logic Utilization	Single Radix approach	Mixed Radix Approach
# MACs	6	2
# Multipliers	6	2
# Adders/Subtractors	24	30
# Registers	2831	1703
# Multiplexers	15	9
# Occupied Slices	1349	832

Table I shows the synthesis results of an 8point FFT using radix 2 and radix 4 blocks. Since the factors are not relatively prime ($\gcd(2,4) \neq 1$), the Cooley- Tukey algorithm was chosen in this case. This involves one intermediate twiddle factor multiplication.

TABLE II: SYNTHESIS RESULTS FOR BASIC BLOCKS FROM RADIX-2 UP TO RADIX-8

Slice Logic Utilization	Radi x 2	Radi x 3	Radi x 4	Radi x 5	Radi x 6	Radi x 7	Radi x 8
Number of Slice LUTs	48	146	258	228	750	985	1215
No of occupied slices	12	53	66	66	198	318	336
Average fan out of nonclock nets	1.94	1.94	2.42	1.95	2.40	2.17	2.78
Number of adders/subtractors	4	14	22	24	70	66	105
No: of multipliers	0	0	0	2	0	9	0
No: of registers/flip flops	96	144	192	240	288	336	384
No: of MACs used	0	4	0	26	8	47	8

Table III shows the synthesis results for computation of the FFT of a 40-point sequence, with 40 being factorized into some possible factor combinations, with the factors ranging between 2 and 8. It can be observed from Table III that use of higher radices yield better performance results as compared to the use of lower radices. It can also be seen that the

ordering of the factors or the different stages in the FFT computation does not have a significant impact on the performance.

The total number of available slices being 69120 in the given device, only 17280 slices were occupied which just above 20% of the entire capacity.

TABLE III: SYNTHESIS RESULTS FOR A 40 POINT FFT WHEN FACTORIZED INTO 3 FACTORS

Slice Logic Utilization	2x4x5	2x5x4	4x2x5	4x5x2	5x2x4	5x4x2
# MACs	26	-	-	26	26	32
# Multipliers	2	31	28	5	5	11
#Adders/Subtractors	68	94	94	68	68	62
# Multiplexers	15	15	15	15	15	15

V. CONCLUSION

An FFT processor design methodology yielding optimal speed-area trade-off is proposed and explored by examining feasible factorization techniques and the use of mixed radix FFT computation techniques. The first advantage of proposed processor over several existing reconfigurable processors is that it can compute the FFT of any N -length sequence, with no restriction of N being a power of 2 or the specific radix used. The proposed design uses the Cooley-Tukey FFT algorithm for FFT computation only in those cases where the factors of the number ' N ' are not relatively prime. The Prime-Factor Algorithm is used to compute the FFT in all other cases.

As a demonstration, a comparison between the two approaches was made. The mixed radix approach used reduced number of multiplications and optimized performance as compared to conventional single-radix processors. The need for zero-padding of sequences so that the sequence length becomes a power of the radix used is also significantly reduced as a mixed-radix approach is used. As an example, synthesis result for the FFT computation of 40-point FFT is shown. In turn, there is no shift in the frequency spectrum when the proposed design is used. The use of the Prime Factor algorithm in those cases where the factors of N are relatively prime eliminates the need for any intermediate twiddle factor multiplications, thus in turn, resulting in a better design.

REFERENCES

- [1] S. L. Yang, K. Zhang, H. Liu, J. Huang, and S. Huang, "An efficient locally pipelined FFT processor," *IEEE transactions on circuits And systems II: Express Briefs*, vol. 53, no. 7, pp. 585-589, July 2006.
- [2] S. S. He and M. Torkelson, "A new approach to pipeline FFT processor," in *Proc. The 10th international conference on Parallel Processing Symposium*, April 1996.
- [3] S. W. Yu and E. E. Swartzlander, "A pipelined architecture for the multidimensional DFT," *IEEE Transactions on Signal Processing*, vol. 49, no. 9, pp. 2096-2102, Sep. 2001.
- [4] J. D. Bruguera and T. Lang, "Implementation of the FFT butterfly with redundant arithmetic," *IEEE Trans. Circuits Syst. II*, vol. 43, pp. 717-723, May 1996.
- [5] Z. B. Gao, H. Chen, and Y. Q. Han, "Design and Implementation of a Pipeline Processor for Various-Size FFTs," *Journal of Beijing Institute of Technology*, vol. 123, pp. 268-271, May 2005.
- [6] C. S. Burrus, "Multidimensional Index Mapping. Connexions Project," *Connexions module: m16326*, Version 1.12, May 4, 2012.
- [7] Z. X. Yang, Y. P. Hu, C. Y. Pan, and L. Yang, "Design of a 3780-Point IFFT Processor for TDS-OFDM," *IEEE Transactions on Broadcasting*, vol. 48, pp. 57-61, March 2002.

- [8] A. Saeed, M. Elbably, G. Abdelfadeel, and M. I. Eladawy, "Efficient FPGA implementation of FFT/IFFT processor," *International Journal of Circuits, Systems and Signal processing*, vol. 3, issue 3, 2009.
- [9] J. Li, F. Liu, T. Long, and E. Mao, *Research on Pipelined R2²SDF FFT*, Beijing Institute of Technology, China.
- [10] Y. T. Lin, P. Y. Tsai, and T. D. Chiueh, "Low-power variable-length fast Fourier transform processor," in *Proc. Comput. Digit. Tech.*, vol. 152, no. 4, pp. 499–506, July 2005.
- [11] T. Y. Sung, H. C. Hsin, and L. T. Ko, "Reconfigurable VLSI Architecture for FFT Processor," *WSEAS Transactions on Circuits and Systems*, vol. 8, issue 6, June 2009.
- [12] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comp.*, vol. 19, pp. 297–301, 1965.
- [13] Y. Chen, Y. W. Lin, Y. C. Tsao, and C. Y. Lee, "A 2.4-Gsample/s DVFS FFT processor for MIMO OFDM communication systems," *IEEE J. Solid-State Circuits*, vol. 43, no. 5, pp. 1260–1273, May 2008.
- [14] I. Urriza, L. A. Barragán, D. Navarro, J. I. Artigas, and O. Lucia, *Word Length Selection Method for Controller Implementation on FPGAs Using the VHDL-2008 Fixed-Point and Floating-Point Packages*, Universidad de Zaragoza, María de Luna 1, 50018 Zaragoza, Spain.

Ramesh Bhakthavatchalu is currently working as an assistant professor at Amrita University, India. He is a member of IEEE. He got his masters in applied electronics from College of Engineering, Chennai, India. Prior to Amrita University, he was working as a corporate application engineer at Syntest Technologies, USA and design engineer in Cirrus Logic, USA. His areas of interest include VLSI Design, testing of VLSI circuits and VLSI signal processing. The other authors in this paper are his graduate students who are pursuing their project on optimized FFT Design.