ISSN 2319 – 2518 www.ijeetc.com
Vol. 2, No. 4, October 2013
© 2013 IJEETC. All Rights Reserved

Research Paper

IMPLEMENTATION OF 64-POINT FFT/IFFT BY USING RADIX-8 ALGORITHM

K Venkata Subba Reddy[1]* and K Bala[1]

*Corresponding Author: K Venkata Subba Reddy, ✉ vsubbareddy05@gmail.com

Fast Fourier Transform (FFT) processing is one of the key procedure in popular Orthogonal Frequency Division Multiplexing (OFDM) communication systems. Structured pipeline architectures, low power consumption, high speed and reduced chip area are the main concerns in this VLSI implementation. In this paper, the efficient implementation of FFT/IFFT processor for OFDM applications is presented. The processor can be used in various OFDM-based communication systems, such as Worldwide interoperability for Microwave access (Wi-Max), Digital Audio Broadcasting (DAB), Digital Video Broadcasting-Terrestrial (DVB-T). We adopt single-path delay feedback architecture. To eliminate the Read Only Memories (ROM's) used to store the twiddle factors, this proposed architecture applies a reconfigurable complex multiplier to achieve a ROM-less FFT/IFFT processor and to reduce the truncation error we adopt the fixed width modified booth multiplier. The three Processing Elements (PE's), Delay-Line (DL) buffers are used for computing IFFT. Thus we consume the low power, lower hardware cost, high efficiency and reduced chip size.

Keywords: FFT/IFFT, Single delay feedback path, OFDM, Pipelined architectures, High power consumption

INTRODUCTION

The Fast Fourier Transform (FFT) and its Inverse Fast Fourier Transform (IFFT) are essential in the field of digital signal processing (DSP), widely used in communication systems, especially in Orthogonal Rrequency Division Multiplexing (OFDM) systems, wireless-LAN, ADSL, VDSL systems and Wi-MAX. Apart from the applications, the system demands high speed of operation, low power consumption, reduced truncation error and reduced chip size. By considering these facts, we proposed the ROM-less processor with Single-path Delay Feedback (SDF) pipeline architecture and modified booth width multiplier. The SDF pipelined architecture is used for the high-throughput in FFT processor. There are three types of pipeline structures; they are Single-path Delay Feedback (SDF), Single-path

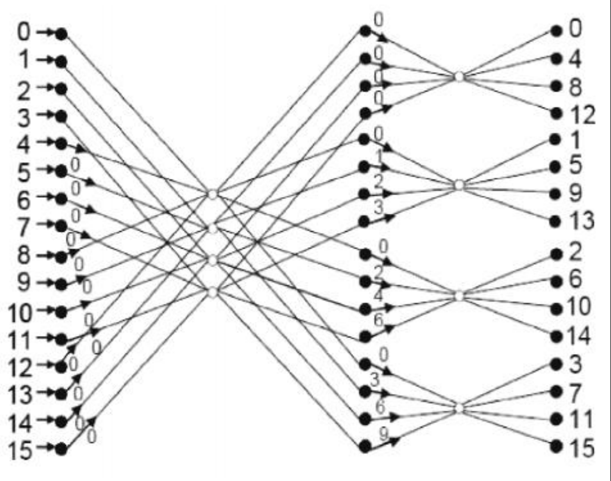
[1] SRTS College of Engineering, Ukkayapalli Road, Kadapa 516002, AP, India.

Delay Commutator (SDC) and Multi-path Delay Commutator. The advantages of Single-path Delay Feedback (SDF) are (1) This SDF architecture is very simple to implement the different length FFT. (2) The required registers in SDF architecture is less than MDC and SDC architectures. (3) The control unit of SDF architecture is easier. We implement the processor in SDF architecture with radix-8 algorithm. There are various algorithms to implement FFT, such as radix-2, radix-4 and split-radix with arbitrary sizes, radix-2 algorithm is the simplest one, but its calculation of addition and multiplication is more than radix-4's. Though being more efficient than radix-2, radix-8 only can process 8 n-point FFT. The radix-8 FFT equation essentially combines three stages of a radix-2 FFT into one, so that one third as many stages are required. Since the radix-8 FFT requires fewer stages and butterflies than the radix 2 FFT, the computations of FFT can be further improved. Using radix-8 algorithm, we propose a 64-point FFT/IFFT processor with ROM architecture.

FFT/ IFFT ALGORITMH

In last three decades, various FFT architectures such as single-memory architecture, dual memory architecture, pipelined architecture, array architecture and cache memory architecture have been proposed. In order to improve the power reduction, we propose a radix-4 64-point pipeline FFT/IFFT processor. In order to speed up the FFT computations, more advanced solutions have been proposed using an increase of the radix. The radix-4 FFT algorithm is most popular and has the potential to satisfy the current need. The radix-4 FFT equation essentially combines two stages of a radix-2 FFT into one, so that half as many stages are required. To calculate 16-point FFT, the radix-2 takes $\log_2 16 = 4$ stages but the radix-4 takes only $\log_4 16 = 2$ stages. A 16-point, radix-4 decimation-in-frequency FFT algorithm is shown in Figure 1. Its input is in normal order and its output is in digit-reversed order. It has exactly the same computational complexity as the decimation-in-time radix-4 FFT algorithm.

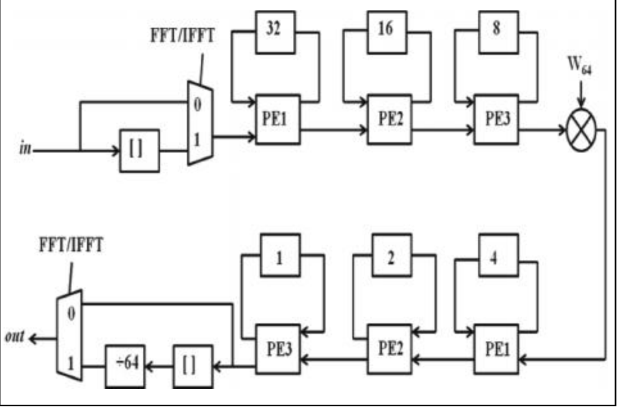
Figure 1: Flow Graph of a 16-Point Radix-4 FFT Algorithm



EXISTING ARCHITECTURE

In this paper, low power techniques are employed for power consumption using reconfigurable complex multiplier. Using radix-4 algorithm, increase the computational speed, further reduce the chip area by three different Processing Elements (PE's) were proposed in this radix-4 64-point FFT/IFFT processor. Our proposed architecture uses a low complexity reconfigurable complex multiplier instead of ROM tables to generate twiddle factors and fixed width modified booth multiplier to reduce the truncation error.

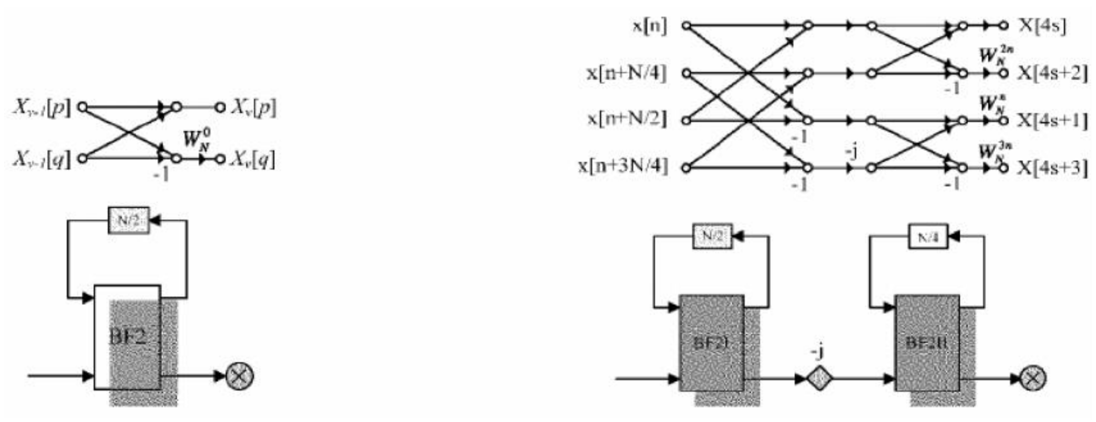
Figure 2: Existing Radix-4 64-Point Pipeline FFT / IFFT Processor



PROPOSED ARCHITECTURE

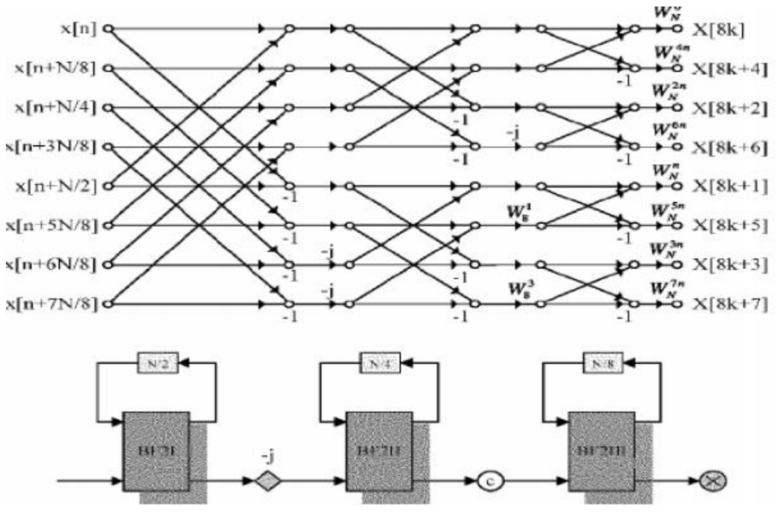
In the R2SDF pipelined architecture, we can find that relationship between the PE numbered and different radix agorithams shown in Figures 3 and 4;◇ presents a complex multiplier,ⓒ denotes a constant multiplier and⊗ denotes a complex multiplier. The radix-2 PE, radix-2^2 PE and radix-2^3 PE respectively map to one time stage of Signal Flow Graph (SFG) of radix-2 butterfly, two time stage of SFG of radix-4 butterfly and

Figure 3: Relationship Between the PE Number and Different Radix Algorithm



(a) **R2SDF PE in Radix-2 Butterfly** (b) **R2^2SDF PE in Radix-4 Butterfly**

Figure 4: R2^3SDF PE in Radix-8 Butterfly



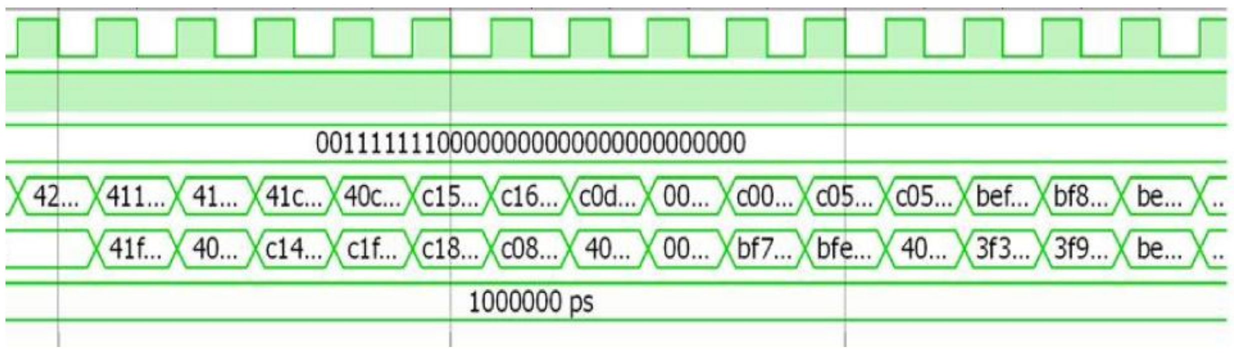
three time stage of SFG of radix-8 butterfly. When R2SDF performs a N-point DFT, where N is a power of 2, every PE of R2SDF includes a complex multiplier except the PE of penultimate time stage employs a "–j" complex multiplier. When the R2^{2}SDF implements N-point DFT, where N is a power of 4, employing PEs o f R2^{2}SDF can completely implement all time stage. Except N isn't a power of 4, adding a extra PE of R2SDF at the last time stage can redeem the R2^{2}SDF to Implement all time stage. When the R2^{3}SDF implements N- point D FT, where N is a power of 8, employing PEs of R2^{3}SDF can completely implement all time stage. Except N isn't a power of 8, adding extra R2SDF PE at the last one remainder time stage or R2^{2}SDF PE at two remainder time stage can completely implement operation of all time stages.

Table 1: Analyze the Number of Complex Multiplications in SDF with Different Radix

R2SDF								
FFT point	64	128	256	512	1024	2048	4096	8192
Constant mul	0	0	0	0	0	0	0	0
Complex mul	4	5	6	7	8	9	10	11
Multiplierarea	131459.2	164324	197188.8	230053.6	262918.4	295783.2	328648	361512.8
R2^{2}SDF								
FFT point	64	128	256	512	1024	2048	4096	8192
Constant mul	0	0	0	0	0	0	0	0
Complex mul	2	3	3	4	4	5	5	6
Multiplierarea	65729.6	98594.4	98594.4	131459.2	131459.2	164324	164324	197188.8
R2^{3}SDF								
FFT point	64	128	256	512	1024	2048	4096	8192
Constant mul	2	2	2	3	3	3	4	4
Complex mul	1	2	2	2	3	3	3	4
Multiplierarea	53275.6	86140.4	86140.4	96345.8	129210.6	129210.6	139416	172280.8

Note: Constant and Complex Multiplier bit width: 20 bit, process: umc. 18. Complex Multiplier: 32864.8 um^2. Constant Multiplier: 10205.4 um^2.

Figure 5: Simulation Results


0011111111000000000000000000000000
42... 411... 41... 41c... 40c... c15... c16... c0d... 00... c00... c05... c05... bef... bf8... be...
41f... 40... c14... c1f... c18... c08... 40... 00... bf7... bfe... 40... 3f3... 3f9... be...
1000000 ps

CONCLUSION

A low power pipelined 64-point FFT/IFFT processor for OFDM applications has been described in this paper.

We present an efficient FFT/IFFT compiler which consists of three IP cores, R2^{3}SDF, R2^{3}MDC and memory-based FFT architectures. The inputs to our developed generator are a set of user-defined parameters. Since our design requires low-cost and consumes low power, as well as reduced SQNR and highly efficient. Hence it can be applied as a powerful FFT/IFFT processor in wireless communication systems.

REFERENCES

1. Bi G and Jones E V (1989), "A Pipelined FFT Processor for Word-Sequential Data", *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. 37, December, pp. 1982-1985.
2. Bouguezel S, Ahmad M O and Swamy M N S (2004), "Improved Radix-4 and Radix-8 FFT Algorithms", *IEEE Transactions on Digital Object Identifier*, Vol. 3, May, pp. 561-564.
3. Cooley J W and Tukey J W (1965), "An Algorithm for the Machine Calculation of Complex Fourier Series", *Math. Comput.*, Vol. 5, No. 5, pp. 87-109.
4. Despain A M (1974), "Fourier Transform Computer Using CORDIC Iterations", *IEEE Trans. Comput.*, Vol. C-23, October, pp. 993-1001.
5. Jia L, Gao Y, Isoaho J and Tenhunen H (1998), "A New VLSI-Oriented FFT Algorithm and Implementation", Proc. Eleventh Annu. IEEE Int. ASIC Conf., pp. 337-341.
6. Oppenheim A V, Schafer R W and John R Buck (1999), *Discrete-Time Signal Processing*, 2nd Edition, Prentice-Hill.
7. Rabiner L R and Gold B (1975), *Theory and Application of Digital Signal Processing*, Prentice-I-Iill, Inc.
8. Takahashi D (2001), "An Extendedsplit-Radix FFT Algorithm", *IEEE Signal Processing Letters*, Vol. 8, No. 5, pp. 145-147.
9. Wold E H and Despain A M (1984), "Pipeline and Parallel Pipeline FFT Processors for VLSI Implementation", *IEEE Trans. Comput.*, Vol. C-33, May, pp. 414-426.