Design of Low Power Mixed Radix FFT Processor for MIMO OFDM Systems

K.Rajendran

Assistant Professor, Aksheyaa College of Engineering, Pulidivakkam. Kanchipuram Dist.

Abstract— **A fast Fourier transform (FFT) is an efficient algorithm to compute the discrete Fourier transform (DFT) and its inverse. Computing a DFT of *N* points in the naive way, using the definition, takes O(N^2) arithmetical operations, while an FFT can compute the same result in only O(*N* log *N*) operations. FFTs are of great importance to a wide variety of applications, from Digital Signal Processing to solving Partial Differential Equations. A pipelined Fast Fourier Transform and its inverse (FFT/IFFT) processor, which utilizes hardware resources efficiently, is proposed for MIMO-OFDM WLAN 802.11n. Compared with a conventional MIMO-OFDM implementation, in which as many FFT/IFFT processors as the number of transmit/receive antennas is used, the proposed architecture uses a single FFT processor with mixed radix and hence reduces hardware complexity without sacrificing system throughput. Further, the proposed architecture can support 1–4 simultaneous input data sequences with sequence lengths of 64 or 128, as needed. A theoretical study and computer simulation (MODELSIM) works of the 128/64 point FFT/IFFT processor is presented.**

Keywords*— *802.11n, Fast Fourier transform (FFT), multiple input multiple-output (MIMO), Orthogonal frequency-division multiplexing (OFDM).

I. INTRODUCTION

The combination of the multiple-input multiple-output (MIMO) signal processing with orthogonal frequency-division multiplexing (OFDM) communication system is considered as a promising solution for enhancing the data rates of the next generation wireless communication systems operating in frequency-selective fading environments. The high throughput task group which establishes IEEE 802.11n standard is going to draw up the next-generation wireless local area network (WLAN) proposal based on the 802.11 a/g which is the current OFDM-based WLAN standards. The IEEE 802.11n standard based on the MIMO OFDM system provides very high data throughput rate from the original data rate 54 Mb/s to the data rate in excess of 600 Mb/s because the technique of the MIMO can increase the data rate by extending an OFDM-based system.

The IEEE 802.11n standard also increases the computational and the hardware complexities greatly, compared with the current WLAN standards. It is a challenge to realize the physical layer of the MIMO OFDM system with minimal hardware complexity and power consumption especially the computational complexity—in VLSI implementation. The FFT/IFFT processor is one of the highest computational complexity modules in the physical layer of the IEEE 802.11n standard. According to IEEE 802.11n standard, the execution time of 128-point and 64-point with 1–4 simultaneous data sequences must be calculated within 3.6 or 4μs. Therefore, if employing the traditional approach to solve the simultaneous multiple data sequences, several FFT/IFFT processors are needed in the physical layer of a MIMO OFDM system. Thus, the hardware complexity of the physical layer in a MIMO OFDM system will be very high. An FFT/IFFT processor with a novel multipath pipelined architecture to deal with the issue of the multiple data sequences for MIMO OFDM applications is proposed. The 128/64-point FFT/IFFT with 1–4 simultaneous data sequences can be supported in our proposed processor with minimal hardware complexity. Furthermore, the power consumption can also be saved by using higher radix FFT algorithm.

II. FFT PROCESSOR FOR MIMO OFDM SYSTEM

In the last three decades, various FFT architectures, such as single –memory architectures, dual memory architecture Pipelined architecture, array architecture and cached – memory architecture, have been proposed. In general, multiple FFT processors are added to deal with multiple data sequences in a MIMO OFDM system and therefore cause a large increase in the hardware complexity and power consumption, compared with that of the single FFT processor.

A good FFT processor should not only provide a high throughput rate, but also deal with multiple data sequences effectively for MIMO OFDM application .In our view, the pipelined architecture should be the best choice for the high throughput rate applications since it can provide high throughput rate with acceptable hardware cost .The pipelined FFT architecture typically falls into one of the two following categories. One is multipath delay commutation feedback (MDC) and the other is single path delay feedback (SDF) . In general the MDC scheme can achieve higher throughput rate by using multiple data paths, While SDF scheme needs less memory and hardware complexity with delay feedback scheme .Besides, the operation of the complex multiplication consumes lots of power in the FFT processor. In order to save power dissipation, higher radix FFT algorithm can be used to reduce the number of complex multiplications. Three-step radix-8 FFT algorithm is chosen in our design to save complex multiplications.

III. ALGORITHMIC FORMULATION

Given a sequences x(n), an N-points discrete Fourier transform (DFT) is defined as

$$X(k) = \sum_{n=0}^{N-1} x[n] \; W_n^{kn} \quad k=0, 1,\ldots 127 \qquad (1)$$

Where x[n] and X (k) are complex numbers.

The twiddle factor is

$$W_n^{kn} = e^{-j\left(\frac{2\pi k}{N}\right)} = \cos\left(\frac{2\pi nk}{N}\right) - j \sin\left(\frac{2\pi nk}{N}\right) \qquad (2)$$

In equation (1.1), the computational complexity is through directly performing the required computation. By using the FFT algorithm, the computational complexity can be reduced. The radix-FFT algorithm can be easily derived from DFT by decomposing the -point DFT into a set of recursively related -point FFT transform, if is a power of . Higher radix FFT algorithm has less number of the nontrivial complex multiplications, compared with the radix-2 FFT algorithm which is the simplest form in all FFT algorithms .In an example for 128-point FFT and 64-point FFT, the number of nontrivial complex multiplications of radix-8 FFT algorithm is 152 and 48, which is only 58.9%and 48.9% of that of radix-2 FFT algorithm. Thus, in order to save power dissipation of the complex multiplier operation, we choose radix-8 FFT algorithm to reduce the number of nontrivial complex multiplications. Because 128-point FFT is not a power of 8, the mixed-radix FFT algorithm, including radix-2 and radix-8 FFT algorithm, is needed.

The algorithm can be briefly described as follows:

First let N=128

n= 64n1+n2, {n1=0.1 and n2=0, 1, …63}

k= k1+2k2, {k1=0.1 and K2=0, 1, …..63} (3)

Using (.3), (1) can be rewritten as,

X (2K2+K1) =

$$\sum_{n2=0}^{63} \sum_{n1=0}^{1} x[64n1+n2] \; W_{128}^{(64n1+n2)(2k2+k1)} \qquad (4)$$

Equation (4) can be considered as a two-dimensional DFT. One is 64-point DFT and the other is 2-point DFT. Then, by decomposing the 64-point DFT into the 8-point DFT recursively 2 times, we can complete the 128-point mixed-radix FFT algorithm. In order to implement a radix-8 FFT algorithm more efficiently, using the radix- FFT algorithm, we further decompose the butterfly of radix-8 FFT algorithm into three steps and apply the radix-2 index map to the radix-8 butterfly.

The radix-2 FFT algorithm is used in the first stage, and the radix-8 algorithm is applied in the second and third stages. And the butterfly of radix-8 in stage 2 and stage 3 can be further decomposed into three steps by using the radix-FFT algorithm.

IV. ARCHITECTURE

A novel 128/64-point FFT/IFFT processor is proposed to support 1-4 simultaneous data sequences for a MIMO OFDM system as shown in Fig. 1 and the relationship of the input and output order of data in each data sequence is shown in Fig. 2 . The proposed FFT architecture consists of module 1, module 2 , module 3, module 4 , conjugate blocks . a division block and multiplexers.

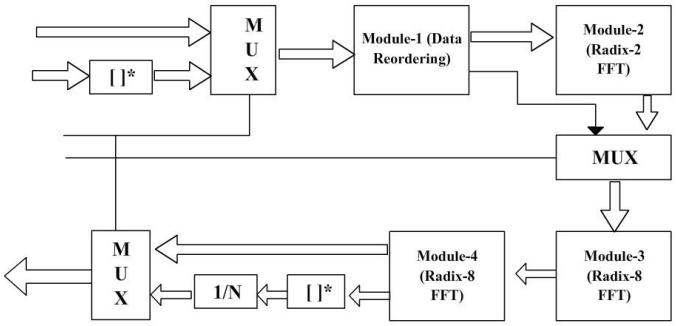

Fig 1. Block Diagram of the proposed 128/64-point FFT/IFFT Processor

A. Module 1

Module 1 contains several different-size delay elements and a switch block, as in Fig.3. The function of Module 1 is to reorder the input sequences to achieve two goals. One goal is to let Module2, Module 3, and Module 4 implement the operation of FFT and IFFT with 1-4 simultaneous data sequences more efficiently. The second goal is to avoid the data sequences in Module 3 to be multiplied by the same twiddle factor in each data path simultaneously.

Thus, the proposed complex multiplier can be used in Module3 to reduce the hardware complexity by using the specified constant multipliers. First, the action of the operation is to separate the four adjoining sequences by one delay unit. Then the separated data will be reordered among four sequences by the appropriate operation of the switch. Finally, the separated data will be adjusted by the delay elements.

The reordered data will be separated into 32 groups or 16 groups for 128- or 64-point, respectively.Each group contains four data sequences, A, B, C, and D. And each data sequence in the same group has the same FFT index. In general, the operation of the FFT is data dependent. In order to implement the multiple data sequences more efficiently, separate the data into several groups. The group replacing the FFT index becomes the basic unit of the FFT operation. For example, group 0 is operated with group 16 in the first stage of the 128-point

mixed-radix FFT algorithm in our design. Because each data sequence in a group has the same FFT index, the operation of FFT/IFFT with multiple data sequences can be regarded as the duplication of the operation of one input sequence in the proposed FFT processor. In each group, the number of operations is decided by the number of data sequences. If there is only one data sequence, the number of operations in the group is one; with two data sequences, the number of operations in the group becomes two and so on. Based on the data ordering and the concept of group in Module 1, the operation of FFT/IFFT with multiple data sequences can be implemented more efficiently.

Input Sequence

A0	A1	A2	A3……..	A124	A125	A126	A127
B0	B1	B2	B3……..	B124	B125	B126	B127
C0	C1	C2	C3……..	C124	C125	C126	C127
D0	D1	D2	D3……..	D124	D125	D126	D127

Output Sequence

A0	B0	C0	D0 ……..……	A31	B31	C31	D31
A64	B64	C64	D64……	A95	B95	C95	D95
A32	B32	C32	D32……	A63	B63	C63	D63
A96	B96	C96	D96……	A127	B127	C127	D127

Fig.2 The order of input and output sequence

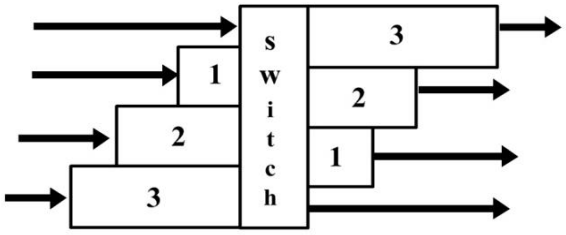

Fig.3 Block Diagram of Module 1

B. Module 2

In general, four complex multipliers are needed in the four parallel approach to implement a radix-2 FFT algorithm. In this work, a group concept is used to deal with multiple data sequences and only two complex multipliers are used in this module. A block diagram of Module 2 consisting of a memory, 4 butterfly units of radix-2 FFT algorithm (BU_2), two complex multipliers, two ROMs, and some multiplexers is shown in Fig.4. If 64-point FFT/IFFT is operated in our scheme, the data will skip this module. The memory size can store the data of 16 groups containing 256 complex data. Because four data paths are adopted in our design, four memory banks are needed to provide four data from memory to BU_2 simultaneously. Only1/8 period of cosine and sine waveforms are stored in ROM and the other period waveforms can be reconstructed by these stored values. The operation of BU_2 is complex addition and complex subtraction from two input data. Because radix-2 FFT algorithm is adopted in this module, period waveforms can be reconstructed by these stored values. The operation of BU_2 is complex addition and complex subtraction from two input data. BU_2 can not start until both input sequences x(n) and x(64+n), when n=0,1,2……63 are available. According to the group data format, the data of the first 16 groups are stored in the memory. When the data of the next 16 groups enter Module 2, the eight input data are loaded into four BU_2s, four from the memory and four from the input, respectively. The operation of BU_2 in a group is dependent on the number of the data sequences.

Four BU_2s generate eight output data according to radix-2 FFT algorithm; four output data will be fed to the Module 3 directly, while the other four output data will be stored in the memory. After the operation of BU_2, there are 16 groups fed to the Module 3 and 16 groups fed back to the memory. Then, these data stored in the memory are read and are multiplied by the twiddle factors simultaneously before they are sent to Module 3. The scheduling of the twiddle factor is shown Fig.5(a)**.** In four-parallel approach, the utilization rate of the complex multiplier is only 50%. The proposed approach can increase the utilization rate and reduce the number of complex multipliers. The detailed operation is described below. Four output data must be generated after BU operation. Two of these four output data are multiplied by the appropriate twiddle factors first before they are stored in the memory; the other two are multiplied by the twiddle factors before they are fed to Module 3. The twiddle factor scheduling of the proposed approach is shown in Fig.5(b).

The data of each data path in the group are multiplied by the same twiddle factors. So the twiddle factor will be unfolded n times, where n is the number of the input data sequences. By rescheduling the timing of the complex multiplications, only two complex multipliers are needed and the utilization of the complex multipliers can achieve 100% in this scheme.

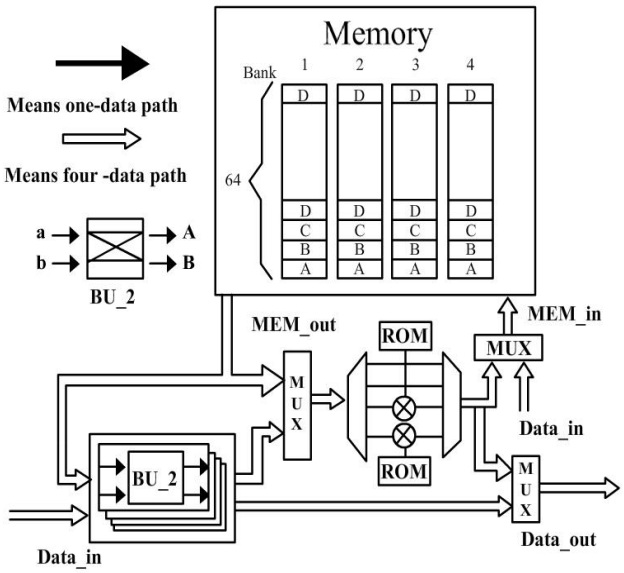

Fig.4 Block diagram of Module2

1st complex multiplier

0 4 8 12 16 20 24 28 32 36 40 44 48 52 56 60

2nd complex multiplier

1 5 9 13 17 21 25 29 33 37 41 45 49 53 57 61

3rd complex multiplier

2 6 10 14 18 22 26 30 34 38 42 46 50 54 58 62

4th complex multiplier

3 7 11 15 19 23 27 31 35 39 43 47 51 55 59 63

Fig. 5(a) Scheduling of twiddle factor where p is from 0 to 63

1St : 0 4 8 12 16 20 24 28 32 36 40 44 48 52 56 60 2 6 10 14 18 22 26 30 34 38 42 46 50 54 58 62

2nd : 1 5 9 13 17 21 25 29 33 37 41 45 49 53 57 61 3 7 11 15 19 23 27 31 35 39 43 47 51 55 59 63

Fig. 5(b) The proposed unfolding approach with two complex multipliers

C. Module 3

Module 3 consists of three stages and one modified complex multiplier, as shown in Fig.6. The architecture of Module 3 is directly mapped from 3-step radix-8 FFT algorithm. Four identical components are used in each stage since four-parallel approach is adopted. In our design, four input data are loaded into Module 3 from the previous module simultaneously. The different data sequences A, B, C, and D are considered as a group, as shown in Fig.6. The size of the delay element in three stages is 32 (8 group), 16 (4 group) and eight (2 group), respectively, as shown in Fig.6. The function of delay element is to store the input data until the other available input data is received for the BU_2 operation.

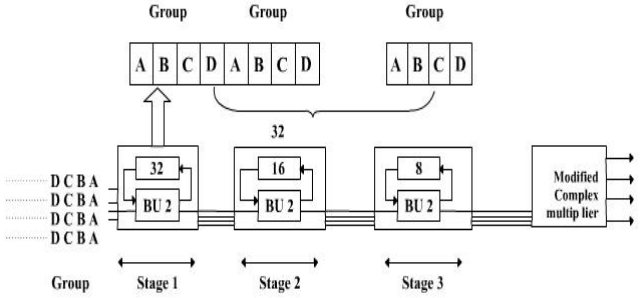

Fig .6 . Block diagram of Module 3

The operations of BU_2 of each stage, which is the same as that of BU_2 in Module 2 shown in Fig.4, are the complex addition and the complex subtraction. For example, the data of the first 8 groups are stored in the delay elements in the first stage of Module 3. When the data of the next groups are valid from the input, eight data are fed to four BU_2s from both the input and the delay element. The output data generated by the BU_2 in the first stage or second stage are multiplied by a trivial twiddle factor before they are fed to the next stage. These twiddle factors can be implemented efficiently. But the four output data from the third stage of Module 3 need to be multiplied by the nontrivial twiddle factors simultaneously in the modified complex multiplier. In the same group, the output data of each data path are multiplied by the same twiddle factors. It is inefficient to build four complex multipliers in four data paths for multiplying different twiddle factors simultaneously. So the proposed approach is to reduce the complexity of the complex multipliers. The twiddle factors of the modified complex multiplier are the real and imaginary parts of the twiddle factor and p is from 0 to 49.

In practice, only nine sets of constant values, with p=0 to 8 are needed because the other twiddle factors can be obtained by using the trigonometric function. And these constant values can be realized more efficiently by using several adders and shifters. The detailed block diagram and function statement is described. The gate count of this approach can be saved about 38%, compared to four-complex-multiplier approach. And the performance of this approach is equivalent to that of the four complex multipliers.

D. Module 4

The block diagram of the Module 4 is shown in Fig.7.Three-step radix-8 FFT algorithm is realized in Module 4. There are also three stages in Module 4 to implement three step radix-8 FFT algorithm, as shown in Fig.7. The scheme of the Module 4 is different from that of Module 3 because the two available data of the BU_2 in the second stage and third stage are in the different data paths. So the structure of Module 4 is modified to ensure that the FFT output data are correct. Some output data generated by the BU_2 in the first stage and second stage are multiplied by the nontrivial twiddle factors before they are fed to the next stage.

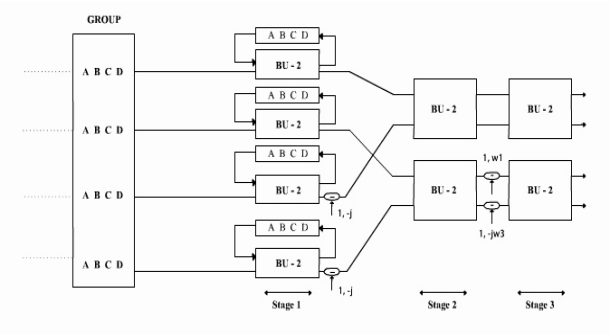

Fig .7 Block diagram of Module 4

V. RESULT AND PERFORMANCE ANALYSIS

The proposed FFT/IFFT processor hardware cost in terms of four-data –sequences 128-point FFT are as follows:
MEMORY SIZE:

- Module 1: 16 words;
- Module 2: 256 words;
- Module 3: 224 words;
- Module 4: 16 words;

Total memory size is 514 words.

- Complex multipliers: 16, where the complexity of modified complex multipliers.
- Complex adders: 48.

In this design, each word of the memory is 8 bits. The word length of the input data in complex multiplier is 8 bits, respectively; the word length of the results is truncated; the word length of complex adders is 8 bits. At the operation clock rate of 161 MHz, the computation of 128-point FFT/IFFT with four data sequences requires approximately 4ns and power consumption 2.26Mw. Due to the execution the functional behavior of the synthesized circuit is checked by using VHDL-XLsimulator.

VI. CONCLUSION

In this design, 64-point and 128-point FFT/IFFT can be supported. Based on the concept of data reordering and grouping, the processor can provide different throughput rates to deal with 1–4 simultaneous data sequences more efficiently. Furthermore, the hardware costs of memory and complex multiplier can be saved by adopting delay feedback and data scheduling approaches. And the number of complex multiplications can be reduced effectively by using higher radix FFT algorithm.

REFERENCES

[1]. S. Magar, S. Shen, G. Luikuo, M. Fleming, and R. Aguilar, "An application specific DSP chip set for 100 MHz data rates," in *Proc. Int.Conf. Acoustics, Speech, and Signal Processing*, vol. 4, Apr. 1988, pp.1989–1992.

[2]. H.Shousheng and M. Torkelson."Designing pipeline FFt processor for OFDM (de) modulation, "in *proc.*URSI Int ,*symp. on signal ,syst,Electron* .., Oct 1998,Vol 29,pp-257-262

[3]. J. O'Brien, J. Mather, and B. Holland, "A 200 MIPS single-chip 1 k FFT processor," in *IEEE Int. Slid-State Circuits Conf. Dig. Tech. Papers*, vol.36, 1989, pp. 166–167.

[4]. B. M. Bass, "A low-power, high-performance, 1024-point FFT processor," *IEEE J. Solid-State Circuits*, vol. 34, no. 3, pp. 380–387, Mar.1999.

[5]. W.-C. Yeh and C.-W. Jen, "High-speed and low-power split-radix FFT," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 51, no. 3, pp.864–874, Mar. 2003.

[6]. Y.W .Lin,H,-Y ,Liu ,and C,-Y ,Lee,"A 1 GS/s FFt/IFFT processor for UWB application ,"*IEEE,J.solid-state circuits*, vol ,40,no.8,pp. 1726-1735,Aug ,2005.

[7]. E.E Swartzlander.W.K.W.Ypung ,and S.J.Joseph. "A radix 4 delay commutator for fast Fourier transform processor implementation. " *IEEE J.Solid –State Circuits*,Vol19,no.10,pp,702-709,Oct .1984.