CORDIC 알고리듬에 기반 한 OFDM 시스템용 8192-Point FFT 프로세서

준회원 박 상 윤*, 정회원 조 남 익*

A 8192-Point FFT Processor Based on the CORDIC Algorithm for OFDM System

Sang Yoon Park* *Associate Member*, Nam Ik Cho* *Regular Member*

요 약

본 논문은 OFDM (Orthogonal Frequency-Division Multiplexing) 시스템용 2K/4K/8K-point 복소 FFT (Fast Fourier Transform) 프로세서의 구조와 그 구현방법을 제안한다. 제안하는 프로세서의 구조는 긴 길이의 DFT를 짧은 길이의 다차원 DFT로 분할하기 위하여 쿨리-투키 알고리듬에 기반 한다. 전치 메모리, 셔플 메모리, 메모리 합성 방법은 다차원 변환을 위한 메모리의 능률적 조작을 위해 사용한다. Booth 알고리듬과 CORDIC (COordinate Rotation DIgital Computer) 프로세서는 각 차원에서 트위들 팩터 곱셈을 위해 사용한다. 또한, CORDIC 프로세서에는 트위들 팩터를 저장하기 위해 필요한 ROM의 사용을 막기 위해 트위들 팩터 발생 방법을 제안한다. 전체 2K/4K/8K FFT 프로세서는 600,000 게이트를 사용하며, 1.8 V, 0.18 μm CMOS를 이용해 구현한다. 제안하는 프로세서는 8K-point FFT를 273 μs마다, 2K-point를 68.26 μs마다 수행할 수 있으며, SNR은 DVB-T의 OFDM을 위해 충분한 48dB를 넘는다.

ABSTRACT

This paper presents the architecture and the implementation of a 2K/4K/8K-point complex Fast Fourier Transform (FFT) processor for Orthogonal Frequency-Division Multiplexing (OFDM) system. The architecture is based on the Cooley-Tukey algorithm for decomposing the long DFT into short length multi-dimensional DFTs. The transposition memory, shuffle memory, and memory mergence method are used for the efficient manipulation of data for multi-dimensional transforms. Booth algorithm and the COordinate Rotation DIgital Computer (CORDIC) processor are employed for the twiddle factor multiplications in each dimension. Also, for the CORDIC processor, a new twiddle factor generation method is proposed to obviate the ROM required for storing the twiddle factors. The overall 2K/4K/8K-FFT processor requires 600,000 gates, and it is implemented in 1.8 V 0.18 μm CMOS. The processor can perform 8K-point FFT in every 273 μs, 2K-point every 68.26 μs at 30MHz, and the SNR is over 48dB, which are enough performances for the OFDM in DVB-T.

I. Introduction

Discrete multitone is widely used for digital data transmission systems such as xDSL and digital video/audio broadcasting. Specifically, the orthogonal frequency-division multiplexing (OFDM) is a multicarrier transmission technique employed in European digital terrestrial video transmission (DVB-T) system[1-2]. OFDM divides the available spectrum into many carriers, each

* 서울대학교 전기공학부 (sanguni@ispl.snu.ac.kr)
논문번호 : 020216-0507, 접수일자 : 2002년 5월 7일

being modulated by a low rate data stream splitted from the original stream. OFDM is similar to the FDMA in that multiple user access is achieved by subdividing available bandwidth into multiple channels. However, OFDM is a more efficient way of multiple access because the channel spacing is much narrower. This is achieved by making all the carriers orthogonal to one another, preventing interference between the closely spaced carriers.

The number of carriers is usually very large in DVB-T (more than 1024), and thus many modulators should be used in parallel, which is realistically unfeasible. Hence, it is implemented by using the IFFT processor of size 2^n that is slightly greater than the number of subcarriers. A simplified block diagram of modulation and demodulation schemes based on the FFT processor is shown in Figure 1. Each synchronous subcarrier generated by the IFFT is modulated by the baseband signal processing that exploits the orthogonal property of the base function of carriers. The IEEE802.11a standard for wireless LAN specifies that only 52 out of 64 possible subcarriers are modulated by the transmitter, and thus single chip for FFT processor is not required due to its short length. On the other hand, OFDM for DVB-T requires baseband signal processor to process about 8000 subcarriers concurrently. Thus a large number of subcarriers is required in OFDM modulation, and high speed and long FFT processor is needed. A development of IC process technolodgy has allowed the single chip to be used to meet the standard, and thus the major advantage of OFDM, single-frequency network (SFN) can also be used. Hence, the FFT processor is one of the most important modules for the implementation of the OFDM system.

This paper presents an FFT processor that can perform 2K and 8K DFT for European standard, and also 4K for Japanese standard. The conventional implementations are focused on 2K/8K points with the Booth algorithm for the multiplication[10-11]. The proposed system can also perform 4K point FFT, and employed CORDIC algorithm

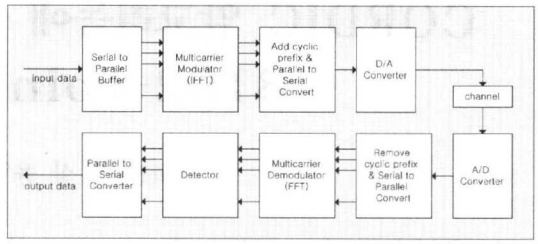


Fig 1. Block diagram of modulation/demodulation schemes in OFDM system.

for the multiplications in order to save memory. The design focuses of the proposed processor are efficient use of memory, ROM size reduction, and high output SNR. In the memory system, transposition and shuffle memory are employed for efficient memory usage and addressing[3-4]. Also, by using the memory mergence method, memory requirement is further reduced. For the complex multiplier, different methods are used depending on the size of the decomposed DFTs. More specifically, the complex multiplication in short-length DFT modules is realized with only two or three real multipliers by using the trigonometric properties, whereas the conventional complex multiplier employs four real multiplications. The CORDIC algorithm is applied to the twiddle factor multiplication for the long length DFT modules[5-8]. Twiddle factor generation method for the CORDIC algorithm is also proposed, which allows additional memory saving.

This paper is organized as follows; The basic building block for the Cooley-Tukey algorithm is given in Section Ⅱ. In Section Ⅲ, the description of overall architecture is given. In Section Ⅳ and Ⅴ, complex multiplier and memory control are presented. Section Ⅵ presents the feature of the implemented chip. Finally, Section Ⅶ gives conclusions.

Ⅱ. Basic Building Block for the Cooley-Tukey Algorithm

The N-point discrete Fourier transform is defined by

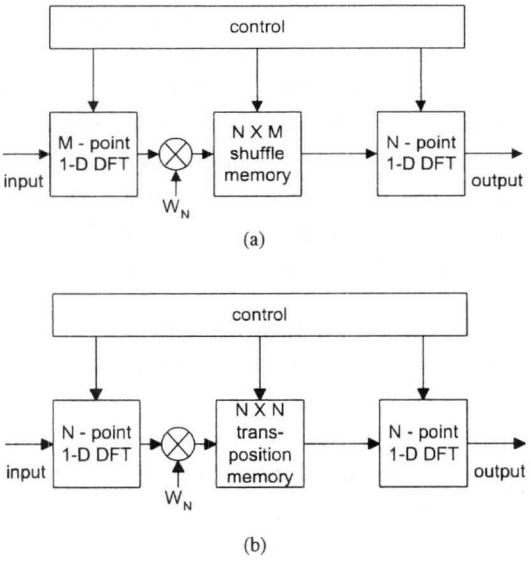


Fig 2. Basic building blocks of the proposed system based on the Cooley-Tukey algorithm (a) using the shuffle memory, (b) using the transposition memory.

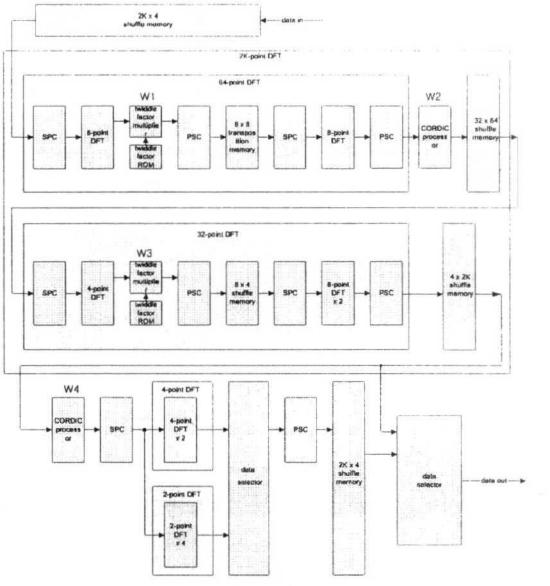


Fig 3. Block diagram of the proposed 2K/4K/8K FFT structure (SPC : Serial to Parallel Converter, PSC : Parallel to Serial Converter.

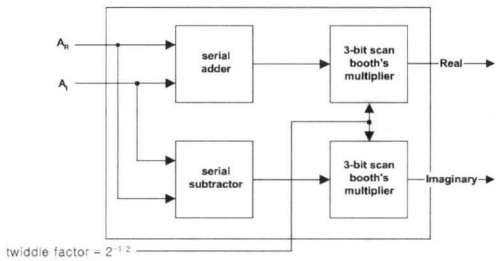
$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, \quad k = 0, 1, \ldots, N-1 \qquad (1)$$

where $W_N = e^{-j(2\pi/N)}$. The input $x(n)$ and the output $X(k)$ are complex numbers. There are many fast algorithms for the implementation of DFT, such as DIT (Decimation-in-Time), DIF (Decimation-in-Frequency), Cooley-Tukey[9], and Winograd algorithm[12-13]. For the 2^n-DFT, Cooley-Tukey algorithm results in DIT or DIF algorithm. In this paper, Cooley-Tukey algorithm is used, i.e., the long-length DFT is decomposed into short-length multi-dimensional transforms.

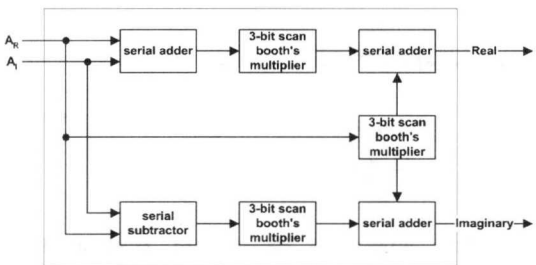
Using the Cooley-Tukey algorithm, one-dimensional input is mapped into a two-dimensional sequence. For example, 16-point input data are first arranged into a two-dimensional 4×4 array. Then, 4-point DFTs are computed in the column order, and appropriate twiddle factors are multiplied to each element of the matrix. 4-point DFT is again applied to each row of the matrix. In summary, the basic building block of the DFT processor based on the Cooley Tukey algorithm is described in Figure 2. As shown in Figure 2, the MN point DFT computation is realized by four modules : M-point row (column) order DFT, twiddle factor multiplier, memory shuffle, and N-point column (row) order DFT. Shuffle and transposition memory are required for pipelining. More specifically, if the M-point DFT module performed computations for row order data, then the N-point DFT module reads the data in the column order. Hence, the M-point DFT should read new data in the column order for the pipelined processing which fully exploits DFT and memory modules. Thus the memory modules should read/write in column and row order alternatively. This is realized by the shuffle memory when $M \neq N$, and transposition memory when $M = N$. By recursively using the basic building block in Figure 2 according to the Cooley-Tukey algorithm, short length DFT modules can be easily extended to 2K, 4K and 8K DFTs.

Ⅲ. Overall Architecture

In this section, the architecture of the proposed FFT processor is described in top-down order. The main components of the proposed architecture are the short-length DFTs, and transposition



(a)



(b)

Fig 4. (a) Complex multiplier with two real multiplications. (b) Complex multiplier with three real multiplications.

memory (TM) or shuffle memory (SM) for interfacing each short-length DFTs. The overall architecture of the proposed 2K/4K/8K FFT processor is shown in Figure 3. As shown in the figure, the 8K-point DFT is decomposed into $2K \times 4$ two-dimensional DFT by the Cooley-Tukey algorithm. In the similar manner, 4K-point DFT is decomposed into $2K \times 2$ DFT, and the 2K-point DFT is again decomposed into 64×32 DFT. They are again decomposed into smaller DFTs such as 8 and 4-point DFT. Hence, the basic elements of the proposed method are the 4 and 8-point DFTs, twiddle factor multipliers (W1, W2, W3, and W4 in Figure 3), and the memory controllers. Note that the TM is used in 64-point DFT, and the SM is used elsewhere, where both are needed for converting the row-column order addressing into column-row order. Detailed descriptions of multipliers and memory systems will be given in the following sections.

Ⅳ. Complex Multiplier

The 4-point and 8-point DFTs are realized by direct implementation of the signal flow graph of decimation-in-time FFT[12]. For the efficient computation of the twiddle factor multiplication, different methods are applied to three different cases : two real multipliers in 8-point DFT, three real multipliers for the 32 and 64-point DFT, and CORDIC multiplier for the higher point DFTs.

4.1 Two Real Multiplications

In 8-point DFT, the only non-trivial twiddle factor is $\cos(0.75\pi) \pm j\sin(0.75\pi)$ (Others are trivial twiddle factors such as ± 1, $\pm j$). Non-trivial twiddle factor multiplications in the 8-point DFT can be expressed as

$$\begin{aligned} A \times W_N^{nk} &= (A_R + jA_I) \times \left(-\frac{1}{\sqrt{2}} \pm j\frac{1}{\sqrt{2}}\right) \\ &= (-A_R \mp A_I)\frac{1}{\sqrt{2}} + j(-A_I \pm A_R)\frac{1}{\sqrt{2}}, \end{aligned} \tag{2}$$

where $A = A_R + jA_I$ is the input to the multiplier, and

$$W_N^{nk} e^{-j\frac{2\pi}{N}nk} = \cos(0.75\pi) \pm j\sin(0.75\pi).$$

Since the real and imaginary parts of the twiddle factor are both $1/\sqrt{2}$, two real multipliers are sufficient in the 8-point DFT module. They are implemented by the Booth algorithm, and the schematic diagram of the multiplier is shown in Figure 4(a).

4.2 Three Real Multiplications

Though the complex multiplication requires four real multiplications, the number of real multipliers can be reduced into three by using the simple properties of trigonometric functions. More specifically, the twiddle factor multiplication can be expressed as

$$\begin{aligned} A \times W_N^{nk} &= (A_R + jA_I) \times (\cos\theta - j\sin\theta) \\ &= (A_R\cos\theta + A_I\sin\theta) \\ &\quad + j(A_I\cos\theta - A_R\sin\theta). \end{aligned} \tag{3}$$

The terms in the righthand of (3) can be expressed as

$$\begin{aligned} A_R\cos\theta + A_I\sin\theta &= A_R(\sin\theta + d) + A_I\sin\theta \\ &= (A_R + A_I)\sin\theta + A_R d, \\ A_I\cos\theta - A_R\sin\theta &= A_I\cos\theta - A_R(\cos\theta - d) \\ &= (A_I - A_R)\cos\theta + A_R d, \end{aligned} \tag{4}$$

where $d = \cos\theta - \sin\theta$. Hence, $A \times W_N^{nk}$ can be realized by using three real multipliers as shown in Figure 4(b). Table 1 compares the proposed complex multiplier with the conventional complex multiplier in the accuracy and hardware complexity. Both are realized by using the Booth algorithm as stated previously. In the simulation, 131,072 combinations of 4096 12-bit integer random inputs and 32 twiddle factor angles at every $1/16\,\pi$ (rad) are used with double float precision arithmetic. The result shows that the proposed algorithm saves hardware by about 23% and yields about 0.007 lower RMS error than the conventional approach.

Table 1. Comparison between proposed complex multiplier and the conventional complex multiplier (Method1 : four real multipliers, Method2 : three real multipliers).

Method		RMS	mean error	abs mean error	abs max error	gate count
Method1	Real	0.3856	-0.0007	0.2945	1.1614	23,589
	Imag	0.3846	-0.0004	0.2932	1.1690	
Method2	Real	0.3782	-0.0010	0.2888	1.1453	18,171
	Imag	0.3785	-0.0008	0.2887	1.1611	

4.3 CORDIC Processor

In the twiddle factor multiplications for larger transforms (W2 and W4 in Figure 3), the Booth multiplier is not efficient since it requires a large ROM for storing many twiddle factors. For example, the conventional multiplier with four real multiplications needs two ROM storages (sine and cosine value) for every twiddle factor, which results in 81,920 bits of ROM in the case of 8K-point FFT. In order to obviate such large ROM, a complex multiplier is employed based on the COordinate Rotation DIgital Computer (CORDIC) algorithm[5].

4.3.1 CORDIC Algorithm

CORDIC is an iterative arithmetic algorithm proposed by Volder in 1959[14]. Using the CORDIC algorithm, elementary functions such as trigonometric, exponential, and logarithm can be evaluated efficiently. The basic function of the CORDIC is to rotate a 2×1 vector in a linear, circular, or hyperbolic coordination systems.

For computing the FFT using CORDIC, the angle of the twiddle factor is decomposed into the weighted sum of the set of predefined elementary rotation angles. That is, the rotation angle θ can be represented as

$$\theta = \sum_{i=0}^{N-1} \mu_i a(i), \tag{5}$$

where N is the number of iterations, and the term μ_i is a sequence of ± 1s which determines the direction of remaining angle. And the $a(i)$ is the i-th elementary rotation angle which can be denoted as

$$a(i) = \tan^{-1}[2^{-i}]. \tag{6}$$

The CORDIC algorithm consists of two parts, namely the iteration process and the scaling factor multiplication process. At each iteration, the input vector is rotated by the angle $a(i)$. At the i-th iteration, the rotation matrix can be described as

$$\begin{aligned} \boldsymbol{P}(i) &= \begin{bmatrix} 1 & -\mu_i 2^{-i} \\ \mu_i 2^{-i} & 1 \end{bmatrix} \\ &= \sqrt{1+2^{-2i}} \begin{bmatrix} \cos a(i) & -\mu_i \sin a(i) \\ \mu_i \sin a(i) & \cos a(i) \end{bmatrix}, \end{aligned} \tag{7}$$

$$z(i+1) = z(i) - \mu_i a(i), \tag{8}$$

where $z(i)$ is the remaining angle after the i-th iteration process. The iteration process relates the output vector $\boldsymbol{v}(i+1) = [x(i+1), y(i+1)]$ to its input vector $\boldsymbol{v}(i) = [x(i), y(i)]$ as

$$\boldsymbol{v}(i+1) = \boldsymbol{P}(i)\,\boldsymbol{v}(i), \quad i = 0, 1, \ldots N-1. \tag{9}$$

However, since the iteration process is not performed on a circle of fixed radius, it is needed to normalize the magnitude of the result

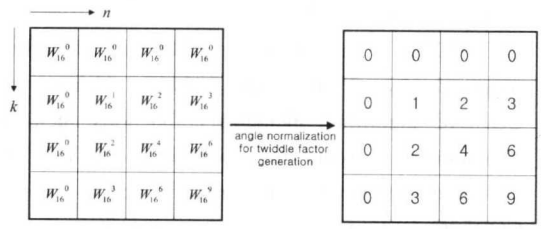


Fig 5. Twiddle factor generation method for the 16-point DFT.

at the final stage. The output of the iteration process $\boldsymbol{v}(N)$ can be divided by the scaling factor given by

$$K(N)=\prod_{i=0}^{N-1}k(i)=\prod_{i=0}^{N-1}\sqrt{1+2^{-2i}}, \tag{10}$$

where $k(i)$ is the scaling factor at the i-th iteration process. Hence, the scaling factor multiplication process is represented as

$$\tilde{\boldsymbol{v}}(N)=\frac{1}{K(N)}\ \boldsymbol{v}(N), \tag{11}$$

where $\tilde{\boldsymbol{v}}(N)$ is the CORDIC output.

4.3.2 Twiddle Factor Generation Method

The conventional CORDIC processor requires ROM for storing the rotation angles of twiddle factors. More specifically, about 3000-word 16-bit ROM and additional hardware are needed for storing the twiddle factors for the proposed system. Hence, this study proposes a twiddle factor generation method that obviates the ROM in W2 and W4 of Figure 3. This method is based on the fact that the angle of the twiddle factor W_N^{nk} is just $\frac{2\pi}{N}nk$, and thus the input index n times the output index k is closely related to the rotation angle. At the iteration process, the angle is updated as

$$z(i+1)=z(i)-\mu_i\tan^{-1}[2^{-i}],\quad i=0,1,\ldots N-1 \tag{12}$$

where $z(0)$ is the angle of the twiddle factor $\frac{2\pi}{N}nk$. Dividing both sides by $N/2\pi$, a normalized angle update

$$z'(i+1)=z'(i)-\mu_i\tan^{-1}[2^{-i}]\frac{N}{2\pi},\quad i=0,1,\ldots N-1 \tag{13}$$

is achieved, where $z'(0)=nk$. Since $\mu_i=\pm1$, the proposed CORDIC processor needs only the constants $\tan^{-1}[2^{-i}]\frac{N}{2\pi}$, which can be implemented using logic. Since n and k are input and output address respectively, they can be easily obtained from address lines, and normalized angle is their product as shown in Figure 5.

Twiddle factor generation method can also be efficiently applied to all other FFT algorithms as well as Cooley-Tukey algorithm. A hardware requirement of the complex multiplier is shown in Table 2. Using the proposed scheme, not only the overall ROM size but also additional hardwares such as word-line decoder and pull up registers, are significantly reduced compared to the conventional approach.

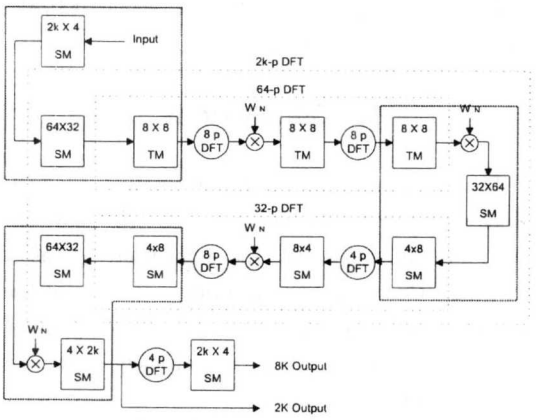
Table 2. Hardware requirements for the complex multiplier (given 16-bit for each sine and cosine value).

	conventional approach (four real multipliers)	Proposed scheme	
		three real multipliers	CORDIC processor
ROM size	81,920 bits	0 bit	0 bit
Gate count (excluding ROM)	23,589 gates	18,17 gates	26,974 gates

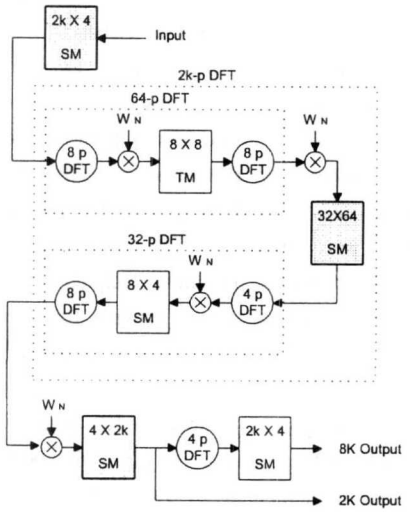
V. Memory Control

5.1 Memory Address Generation Method

For the implementation of DFT using the Cooley-Tukey algorithm, memory control also plays an important role. Transposition memory (TM) and shuffle memory (SM) is employed in [4], which are based on the memory address generation methods that convert row order to column order with a minimum amount of memory. For the N × N transforms, a conventional TM can be used. But, for the M × N transforms, where M is not equal to N, SM provides a memory-efficient architecture. SM and TM have



(a)



(b)

Fig 6. The simplified global structure of the proposed FFT processor (a) before using memory mergence method, (b) after using memory mergence method.

the following characteristics :

A. SM and TM can receive an input array in row major order while providing an output array in column major order.

B. SM and TM can support 'read-then-write at the same address operation.'

In the proposed system, the transposition memory is used in 64-point DFT, and the shuffle memory are used elsewhere as shown in Figure 3. Further details about memory address generation can be found in [3-4].

FRONT 2K x 4 ADDRESS SEQUENCE

Line 1 :	12	11	10	9	8	7	6	5	4	3	2	1	0	
Line 2 :	10	9	8	7	6	5	4	3	2	1	0	12	11	2Kx4
Line 3 :	5	4	3	2	1	0	10	9	8	7	6	12	11	64x32
Line 4 :	2	1	0	5	4	3	10	9	8	7	6	12	11	8x8

Fig. 7 An example of the memory mergence method.

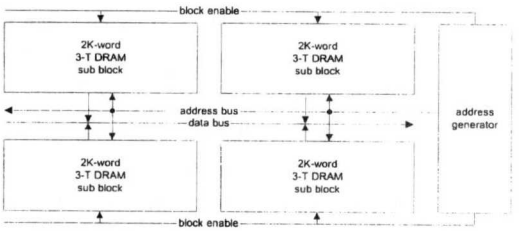


Fig. 8 Structure of the 8K-word shuffle memory.

5.2 Memory Mergence Method

Figure 6(a) is a simplified global structure of the proposed FFT processor. As shown in the figure, three transposition or shuffle memories are used to compute one FFT module. For example, three 8×8 transposition memories are used in 64-point FFT. But, since one of the memories is larger than the other two memories in the same box, three memories are redundant. They can be reduced to only one memory after using the memory mergence method. Figure 7 shows an example of memory mergence method. Line 1 in Figure 7 shows the indices of memory address of the first box in Figure 6(a). If the positions of the most significant 2 bits are replaced with that of other 11 bits as Line 2, front 8K memory can play the role of $2K \times 4$ shuffle memory. In the similar manner, if the positions of the most significant 5 bits are replaced with that of other 6 bits except the 2 bits of Line 1, front 8K memory can play the role of not only $2K \times 4$ but also 64×32 shuffle memory as Line 3. Finally, performing the operation of Line 4, three memories in the box can be substituted by one 8K memory. Figure 6(b) shows a global structure after performing a memory mergence method.

5.3 Implementation of Memory System

Basic building blocks of SM are memory-cell-array using 3-T DRAM, address generator,

word-line decoder, input data buffer, sense amplifier, and bus driver. In particular, because bit line of read mode can be operated independent of that of write mode in the memory-cell-array using 3-T DRAM, consecutive read-then-write at the same address can be performed with partitioning clock cycle. The 8K-word shuffle memory in Figure 8 is decomposed into four 2K-word memory blocks, and 2K-word memory sub block is again decomposed into eight 256-word 3-T DRAM for the stability of read mode. 8K-word shuffle memory operates as 8K or 2K-word memory according to the block selection signal. Moreover, the proposed memory system has been designed to tolerate guard interval that is defined in OFDM system in order to deal with the delay-spread distortion of the transmitted data.

Table 3. The chip features of the proposed system.

	2K-point FFT	4K-point FFT	8K-point FFT
Time delay	212.27 μs	486.67 μs	896.27 μs
Speed	68.27 μs	136.5 μs	273 μs
SNR	49.0826dB	49.1275dB	48.5025dB

Ⅵ. Chip Features

Based on the fixed-point error analysis of the CORDIC for DFT[15-17], the finite precision rounding and approximation error of the proposed system were analyzed, and the appropriate word length was determined. More specifically, the proposed system has been designed with $2 \times$ 10-bit input, 2×12-bit output, and 2×16-bit internal precision. Also, 5 extra bits are added, and 17 iteration processes have been performed in order to guarantee the accuracy of the conventional complex multiplier. Table 3 shows the chip features of the proposed system. The performance of the proposed FFT processor always exceeds the SNR of 48dB, and which is enough for digital terrestrial TV broadcasting. The proposed architecture has been synthesized in the 1.8 V 0.18 μm CMOS technology. It results in about 600,000 gates for logic and storage.

Ⅶ. Conclusions

This study designed and implemented an FFT processor which can perform 2K, 4K and 8K DFT for OFDM system. The architecture is based on the Cooley-Tukey algorithm for modular structure, and thus the 2K/4K/8K point selection is easily achieved. For an efficient complex multiplication, a complex multiplier was used which requires two or three real multipliers rather than four as in the conventional methods. In addition, complex multiplier based on the CORDIC algorithm is used and a twiddle factor generation method is proposed for saving the ROM size. A transposition memory for the pipelined processing, and also the shuffle memory, which is a generalization of the transposition memory, were used. They receive inputs in row major order while providing outputs in column major order. Thus they can provide for a memory efficient architecture for separable 2-D transforms. Also, by using the memory mergence method, the memory usage is increased. The overall 2K/4K/8K-FFT processor requires 600,000 gates, and it is implemented in 1.8 V 0.18 μm CMOS. The processor can perform 8K-point FFT every 273 μs, and 2K-point every 68.26 μs at 30MHz, which exceeds the OFDM symbol rate. The output SNR is higher than 48dB, which is sufficient for DVB-T.

REFERENCES

[1] R. Lassalle and M. Alard, "Principles of modulation and channel coding for digital broadcasting for mobile receivers," *EBU Technical Review*, No. 224, pp. 168-190, Aug. 1987.

[2] W. Y. Zou and Y. Wu, "COFDM: An Overview," *IEEE Trans. on Broadcasting*, vol. 41, No. 1, pp. 1-8, Mar. 1995.

[3] L. J. D'Luna, W. A. Cook, R. M. Guidash, G. W. Brown, T. J. Tredwell, J. R. Fischer, and T.

Tarn, "An 8 × 8 discrete cosine transform chip with pixel rate clocks," *The 3rd Annual IEEE ASIC Seminar and Exhibit*, pp. P7/5.1-P7/5.4, 1990.

[4] K. Kim, "Shuffle memory system," *13th Int. Symp. on Parallel and Distributed Processing*, pp. 268-272, Apr. 1999.

[5] Y. H. Hu, "CORDIC-based VLSI architectures for digital signal processing," *IEEE Signal Processing Mag.*, vol. 9, No. 3, pp. 16-35, July 1992.

[6] R. Sarmiento, V. D. Armas, J. F. Lopez, J. A. Montiel-Nelson, and A. Nunez, "A CORDIC processor for FFT computation and its implementation using gallium arsenide technology," *IEEE Trans. on VLSI Systems*, vol. 6, No. 1, pp. 18-30, Mar. 1998.

[7] A. M. Despain, "Fourier transform computers using CORDIC iterations," *IEEE Trans. on Computers*, vol. C-23, No. 10, pp. 993-1001, Oct. 1974.

[8] ----------, "Very fast Fourier transform algorithms for hardware implementation," *IEEE Trans. on Computers*, vol. C-28, No. 5, pp. 333-341, May 1979.

[9] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Mathematics of Computation*, vol. 19, No. 90, pp. 297-301, Apr. 1965.

[10] E. Bidet, D. Castelain, C. Joanblanq, and P. Senn, "A fast single-chip implementation of 8192 complex point FFT," *IEEE Journal of Solid-State Circuits*, vol. 30, No. 3, pp. 300-305, Mar. 1995.

[11] S. H. Park, D. H. Kim, D. S. Han, K. S. Lee, S. J. Park, and J. R. Choi, "Sequential design of a 8192 complex point FFT in OFDM receiver," *The First IEEE Asia Pacific Conference*, pp. 262-265, Aug. 1999.

[12] A. V. Oppenheim and R. W. Schafer, *Discrete-time Signal Processing*, Englewood Cliffs, NJ : Prentice Hall, 1989.

[13] S. Winograd, "On computing the discrete Fourier transform," *Mathematics of Computation*, vol. 32, No. 141, pp. 175-199, Jan. 1978.

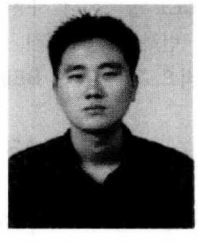
[14] J. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. on Electronic Computers*, vol. EC-8, No. 3, pp. 330-334, Sept. 1959.

[15] M. Bekooij, J. Huisken, and K. Nowak, "Numerical accuracy of fast Fourier transforms with CORDIC arithmetic," *Journal of VLSI Signal Processing*, vol. 25, No. 2, pp. 187-193, June 2000.

[16] Y. H. Hu, "The quantization effects of the CORDIC algorithm," *IEEE Trans. on Signal Processing*, vol. 40, No. 4, pp. 834-844, Apr. 1992.

[17] T. Thong and B. Liu, "Fixed-point fast Fourier transform error analysis," *IEEE Trans. on Acoust., Speech, Signal Processing*, vol. ASSP-24, No. 6, pp. 563-573, Dec. 1976.

박 상 윤(Sang Yoon Park) 준회원



2000년 2월 : 서울대학교 전기공학부 졸업
2002년 2월 : 서울대학교 전기컴퓨터공학부 석사
2002년 2월~현재 : 서울대학교 전기컴퓨터공학부 박사과정
<주관심 분야> 영상신호처리, VLSI 신호처리

조 남 익(Nam Ik Cho) 정회원

1986년 2월 : 서울대학교 제어계측공학과 졸업
1988년 2월 : 서울대학교 제어계측공학과 석사
1992년 2월 : 서울대학교 제어계측공학과 박사
1994년~1998년 : 서울시립대학교 조교수
1999년~현재 : 서울대학교 전기컴퓨터공학부 조교수/부교수
<주관심 분야> 신호처리, 영상신호처리, VLSI 신호처리, 적응신호처리