A 1-GS/s FFT/IFFT Processor for UWB Applications

Yu-Wei Lin, Hsuan-Yu Liu, and Chen-Yi Lee, *Member, IEEE*

***Abstract*—In this paper, we present a novel 128-point FFT/IFFT processor for ultrawideband (UWB) systems. The proposed pipelined FFT architecture, called mixed-radix multipath delay feedback (MRMDF), can provide a higher throughput rate by using the multidata-path scheme. Furthermore, the hardware costs of memory and complex multipliers in MRMDF are only 38.9% and 44.8% of those in the known FFT processor by means of the delay feedback and the data scheduling approaches. The high-radix FFT algorithm is also realized in our processor to reduce the number of complex multiplications. A test chip for the UWB system has been designed and fabricated using 0.18-μm single-poly and six-metal CMOS process with a core area of 1.76×1.76 mm^2, including an FFT/IFFT processor and a test module. The throughput rate of this fabricated FFT processor is up to 1 Gsample/s while it consumes 175 mW. Power dissipation is 77.6 mW when its throughput rate meets UWB standard in which the FFT throughput rate is 409.6 Msample/s.**

***Index Terms*—Fast Fourier transform (FFT), orthogonal frequency division multiplexing (OFDM), ultrawideband (UWB).**

I. Introduction

ULTRAWIDEBAND (UWB) communication systems, which enable one to deliver data from a rate of 110 Mb/s at a distance of 10 m to a rate of 480 Mb/s at a distance of 2 m in realistic multipath environment while consuming very little power and silicon area, are currently the focus of research and development of wireless personal area networks (WPANs). Orthogonal frequency division multiplexing (OFDM) is considered as the leading choice by the 802.15.3a standardization group for use in establishing a physical-layer standard for UWB communications [1]. OFDM-based UWB not only has reliably high-data-rate transmission in time-dispersive or frequency-selective channels without having complex time-domain channel equalizers but also can provide high spectral efficiency. However, because the data sampling rate from the analog-to-digital converter to the physical layer is up to 528 Msample/s or more, it is a challenge to realize the physical layer of the UWB system—especially the components with high computational complexity—in VLSI implementation. The FFT/IFFT processor is one of the modules having high computational complexity in the physical layer of the UWB system, and the execution time of the 128-point FFT/IFFT in UWB system is only 312.5 ns. Therefore, if employing the traditional approach, a great deal of power consumption and high hardware cost of the FFT/IFFT processor will be needed to meet the strict specifications of the UWB system. Thus, this paper proposes a FFT/IFFT processor with a novel multipath pipelined architecture for high-throughput-rate applications. The power consumption and hardware cost can also be saved in our processor by using the higher radix FFT algorithm and less memory and complex multipliers.

This paper is organized as follows. Section II identifies the problems of the implementation of the FFT/IFFT processor in UWB system. Section III describes the 128-point mixed-radix FFT algorithm, including the radix-2 FFT algorithm and the radix-8 FFT algorithm, and the IFFT algorithm. Section IV focuses on describing the proposed FFT/IFFT architecture and compares its hardware cost and throughput rate with some existing FFT architectures in 128-point FFT. Some simulation results are shown in Section V. In Section VI, the microphotograph of the fabricated FFT/IFFT processor and the measurement results are presented. Then, conclusions are drawn in Section VII.

II. Design Issue of the FFT Processor for the OFDM-Based UWB System

A block diagram of the proposed physical layer of OFDM-based UWB system is shown in Fig. 1. It contains a convolutional encoder, a Viterbi decoder, a pilot insertion, a QPSK-modulator/demodulator, a spreading/de-spreading, a guard interval insertion/removal, a 128-point FFT/IFFT, a serial-to-parallel (S/P) converter/parallel-to serial (P/S) converter, an analog-to-digital converter (ADC), a digital-to-analog converter (DAC), and a synchronization and channel estimation block. In the UWB system, the data rate is from 53.3 to 480 Mb/s with code rates $1/3, 11/32, 1/2, 5/8$, and $3/4$. The bandwidth of the transmitted signal is 528 MHz and the OFDM symbol duration is 312.5 ns, including 60.61 ns for cyclic prefix duration and 9.47 ns for guard interval duration [1]. Thus, an FFT/IFFT has to compute one OFDM symbol within 312.5 ns and the throughput rate of this specification in 128-point FFT/IFFT is up to 409.6 Msample/s. In order to implement the physical layer of the UWB system more efficiently, the four-data-path approach is adopted to reduce the data sampling rate from the ADC, as shown in Fig. 1, so that, after the S/P converter, the data sampling rate of each path can be down to 132 Msample/s.

Various FFT architectures, such as single-memory architecture, dual-memory architecture [2], pipelined architecture [3], array architecture [4], and cached-memory architecture [5], have been proposed in the last three decades. In our view, the pipelined architecture should be the best choice for 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

Manuscript received December 6, 2004; revised April 26, 2005. This work was supported in part by the National Science Council of Taiwan, R.O.C., under Grant NSC93-2220-E-009-003, the Ministry of Economic Affairs, R.O.C., under Grant 93-EC-17-A-03-S1-0005, and Media Tek, Inc.

The authors are with the Department of Electronics Engineering, National Chiao University, Hsinchu 300, Taiwan, R.O.C. (e-mail: ywlin@si2lab.org).

Digital Object Identifier 10.1109/JSSC.2005.852007

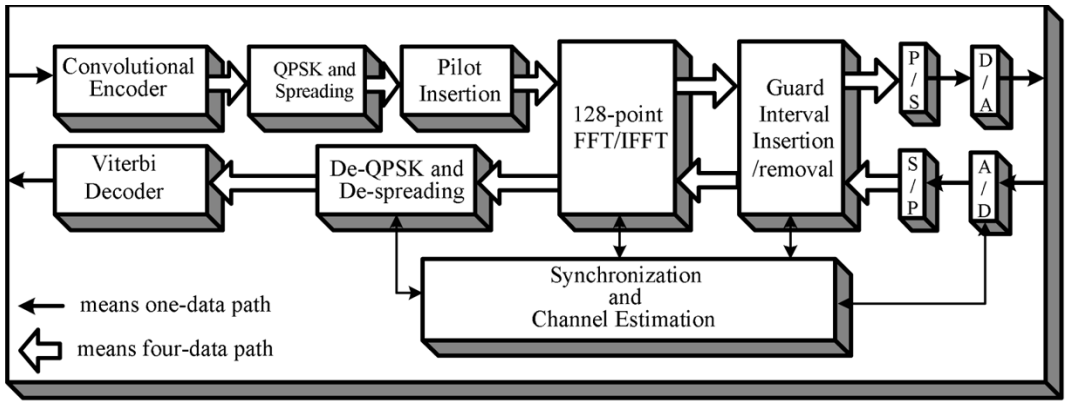


Fig. 1. Block diagram of the physical layer of OFDM-based UWB system.

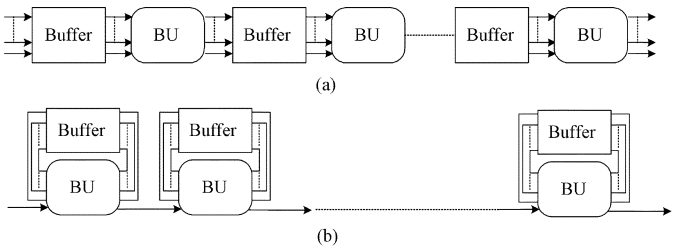


Fig. 2. (a) MDC scheme. (b) SDF scheme.

categories. One is multipath delay commutator (MDC) and the other is single-path delay feedback (SDF), as shown in Fig. 2(a) and (b), respectively [3]. In general, if appropriately reordered M parallel input data can be supported simultaneously in the MDC scheme, this scheme provides M times throughput rate of the SDF scheme. However, there are some limitations on the number of data path, the FFT size, and the radix-r FFT algorithm in the MDC architecture. Besides, the requirement of the memory and complex multiplier in the MDC scheme is more than that of the SDF scheme. In general, the MDC scheme can achieve a higher throughput rate, while the SDF scheme needs less memory and hardware cost.

For high-throughput-rate applications, the MDC architecture is more suitable than the SDF architecture in UWB applications if the input data are reordered in the input buffer before they are loaded into the MDC processor. Unfortunately, the traditional R2 (Radix-2) MDC architecture cannot provide the available throughput rate unless it raises the work frequency [6]; the R4 (Radix-4) MDC architecture, which needs a power of four, has the limitation on FFT size [6], and the Split-Radix (SR) MDC has higher hardware cost [7]. In addition, the higher radix FFT algorithm is difficult to be implemented in the traditional MDC architecture. In general, the higher throughput rate of the FFT processor can be provided by increasing the number of data paths in the MDC pipelined architecture. However, the hardware cost is also increased significantly because more memory and complex multipliers are needed to allow multiple data to be operated simultaneously. The main motivation of this paper is to design a novel four-data-path pipelined FFT architecture, which is called mixed-radix multipath delay feedback (MRMDF), by combining the features of the SDF and MDC architectures. The proposed FFT/IFFT processor not only suits the proposed UWB physical layer, as shown in Fig. 1, but also can provide an available throughput rate to meet the UWB specifications. The MRMDF architecture has lower hardware cost compared with the traditional MDC approach and adopts the high-radix FFT algorithm to save power dissipation. The approach will be described in more detail in Section III–IV.

III. Algorithm

Given a sequence $x(n)$, an N-point discrete Fourier transform (DFT) is defined as

$$X(k) = \sum_{n=0}^{N} x(n) W_N^{kn} \quad k = 0 \ldots 127 \tag{1}$$

where $x(n)$ and $X(k)$ are complex numbers. The twiddle factor is

$$W_N^{nk} = e^{-j(2\pi nk/N)} = \cos(2\pi nk/N) - j\sin(2\pi nk/N). \tag{2}$$

In (1), the computational complexity is $O(N^2)$ through directly performing the required computation. By using the FFT algorithm, the computational complexity can be reduced to $O(N \log_r^N)$, where r means the radix-r FFT. The radix-r FFT algorithm can be easily derived from the DFT by decomposing the N-point DFT into a set of recursively related r-point FFT transform, if $x(n)$ is power of r. In general, higher-radix FFT algorithm has less number of complex multiplications compared with radix-2 FFT algorithm which is the simplest form in all FFT algorithms. In an example, for the 128-point FFT, the number of nontrivial complex multiplications of the radix-8 FFT algorithm is 152, which is only 59.3% of that of the radix-2 FFT algorithm [8]. Thus, in order to save the number of complex multiplications, we choose the radix-8 FFT algorithm. Because the 128-point FFT is not a power of 8, the mixed-radix FFT algorithm, including radix-2 and radix-8 FFT algorithms, is needed. This will be derived in detail below. First let

$$\begin{aligned} &N = 128 \\ &n = 64n_1 + n_2, \quad \begin{cases} n_1 - 0, 1 \\ n_2 = 0 \ldots 63 \end{cases} \\ &k = k_1 + 2k_2, \quad \begin{cases} k_1 = 0, 1 \\ k_2 = 0 \ldots 63. \end{cases} \end{aligned} \tag{3}$$

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

$$\begin{aligned} X(2k_2+k_1) &= \sum_{n_2=0}^{63}\sum_{n_1=0}^{1} x(64n_1+n_2)W_{128}^{(64n_1+n_2)(2k_2+k_1)} \\ &= \underbrace{\sum_{n_2=0}^{63}\left\{\underbrace{\sum_{n_1=0}^{1} x(64n_1+n_2)W_2^{n_1k_1}}_{\text{2 point DFT}}\underbrace{W_{128}^{n_2k_1}}_{\text{twiddle factor}}\right\}W_{64}^{n_2k_2}}_{\text{64 point DFT}} \\ &= \sum_{n_2=0}^{63} BU_2(k_1,n_2)W_{64}^{n_2k_2}. \end{aligned} \tag{4}$$

Equation (4) can be considered as a two-dimensional (2-D) DFT. One is a 64-point DFT and the other is a two-point DFT. Then, by decomposing the 64-point DFT into the eight-point DFT recursively two times, we can complete the 128-point mixed-radix FFT algorithm. In order to implement radix-8 FFT algorithm more efficiently, using the radix-2^3 FFT algorithm proposed by He and Torkelson [3], we further decompose the butterfly of the radix-8 FFT algorithm into three steps and apply the radix-2 index map to the radix-8 butterfly. By a three-dimensional (3-D) linear index map, n_2 and k_2 can be defined as

$$\begin{aligned} n_2 &= 32\alpha_1+16\alpha_2+8\alpha_3+\alpha_4 \\ \alpha_1,\alpha_2,\alpha_3 &= 0,1; \quad \alpha_4 = 0\ldots7. \\ k_2 &= \beta_1+2\beta_2+4\beta_3+8\beta_4 \\ \beta_1,\beta_2,\beta_3 &= 0,1; \quad \beta_4 = 0\ldots7. \end{aligned} \tag{5}$$

By means of (5), (4) takes the following form:

$$\begin{aligned} X(2(\beta_1+2\beta_2+4\beta_3+8\beta_4)+k_1) &= \sum_{\alpha_4=0}^{7}\sum_{\alpha_3=0}^{1}\sum_{\alpha_2=0}^{1}\sum_{\alpha_1=0}^{1} BU_2(k_1,32\alpha_1+16\alpha_2+8\alpha_3+\alpha_4) \\ &\quad\times W_{64}^{(32\alpha_1+16\alpha_2+8\alpha_3+\alpha_4)(\beta_1+2\beta_2+4\beta_3+8\beta_4)}. \end{aligned} \tag{6}$$

The twiddle factor in (6) is

$$\begin{aligned} W_{64}^{(32\alpha_1+16\alpha_2+8\alpha_3+\alpha_4)(\beta_1+2\beta_2+4\beta_3+8\beta_4)} &= W_2^{\alpha_1\beta_1}W_4^{\alpha_2\beta_1}W_2^{\alpha_2\beta_2}W_8^{\alpha_3(\beta_1+2\beta_2)}W_2^{\alpha_3\beta_3} \\ &\quad\times W_{64}^{\alpha_4(\beta_1+2\beta_2+4\beta_3)}W_8^{\alpha_4\beta_4}. \end{aligned} \tag{7}$$

Using (7), (6) becomes

$$\begin{aligned} X(2(\beta_1+2\beta_2+4\beta_3+8\beta_4)+k_1) &= \sum_{\alpha_4=0}^{7} \mathrm{BU}_8(k_1,\beta_1,\beta_2,\beta_3,\alpha_4)W_8^{\alpha_4\beta_4} \end{aligned} \tag{8}$$

where $\mathrm{BU}_8(k_1,\beta_1,\beta_2,\beta_3,\alpha_4)$ is defined as shown in the equation (9) at the bottom of the page.

In (9), we use the radix-2 index map to divide the eight-point DFT into three steps. Fig. 3 shows the signal flow graph (SFG) of the 128-point mixed-radix FFT algorithm. In Fig. 3, the 128-point FFT is divided by three stages, where radix-2 FFT algorithm is used in the first stage, and the radix-8 algorithm is applied in the second and third stages. The black point shown in Fig. 3 means that the twiddle factor will be multiplied at that point, and each butterfly of the radix-8 algorithm is further decomposed into three steps. The twiddle factors, W_8^1 and W_8^3, at the first step are trivial complex multiplications, because they can be written as $\sqrt{2}/2(1-j)$ and $-(\sqrt{2}/2(1+j))$, respectively. Thus, a complex multiplication with one of the two coefficients can be computed using additions and a real multiplication, whose hardware can be realized by six shifters and four adders [9].

The IFFT of an N-point sequence $X(k), k=0,1,\ldots,N-1$ is defined as

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)W^{-nk}. \tag{10}$$

If we take the complex conjugate of (10) and multiply both sides by N, we find

$$Nx^*(n) = \sum_{k=0}^{N-1} X^*(k)W^{nk}. \tag{11}$$

The right-hand side of (11) is recognized to be the FFT of the sequence $X*(k)$ and can be computed using any FFT algorithm [6]. By taking the complex conjugate of (11) and dividing both sides by N, the desired output sequence $x(n)$ is given by

$$x(n) = \frac{1}{N}\left\{\sum_{k=0}^{N-1} X^*(k)W^{nk}\right\}^*. \tag{12}$$

According to (12), the IFFT can be performed by taking the complex conjugate of the incoming data first and then the outgoing data without changing any coefficients in the original FFT

$$\begin{aligned} &\mathrm{BU}_8(k_1,\beta_1,\beta_2,\beta_3,\alpha_4) \\ &\quad= \sum_{\alpha_3=0}^{1}\sum_{\alpha_2=0}^{1}\sum_{\alpha_1=0}^{1}\underbrace{\left\{\underbrace{\underbrace{\mathrm{BU}(k_1,\alpha_1,\alpha_2,\alpha_3,\alpha_4)W_2^{\alpha_1\beta_1}W_4^{\alpha_2\beta_1}}_{\text{1st step}}W_2^{\alpha_2\beta_2}W_8^{\alpha_3(\beta_1+2\beta_2)}}_{\text{2nd step}}W_2^{\alpha_3\beta_3}W_{64}^{\alpha_4(\beta_1+2\beta_2+4\beta_3)}\right\}}_{\text{3th step}} \end{aligned} \tag{9}$$

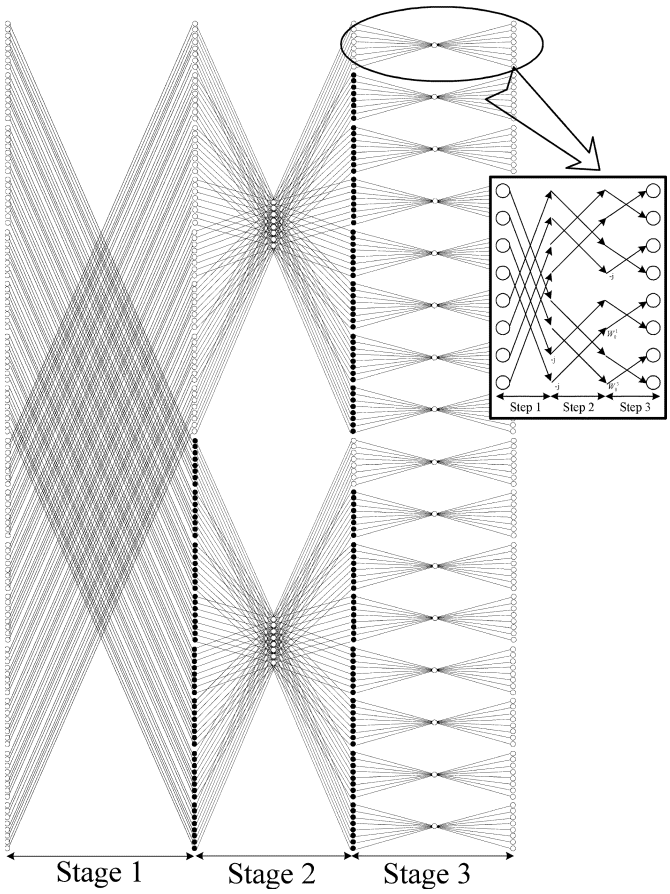


Fig. 3. SFG of the 128-point mixed-radix FFT algorithm.

algorithm so that the hardware implementation can be more efficient.

IV. Architectrue

A. Proposed FFT Architecture for the UWB System

The block diagram of the proposed 128-point FFT/IFFT processor derived from (4), (8), (12), and the SFG (Fig. 3) is depicted in Fig. 4. The proposed MRMDF architecture combining the features of the SDF and MDC architectures consists of Module 1, Module 2, Module 3, conjugate blocks, a division block, and multiplexers. The features of the proposed MRMDF architecture are the following:

- higher throughput rate can be provided by using four parallel data paths;
- the minimum memory is required by using the delay feedback approach to reorder the input data and the intermediate results of each module;
- the 128-point mixed-radix FFT/IFFT algorithm is implemented to save power consumption;
- the number of complex multiplier is minimized by using the scheduling scheme and the specified constant multipliers.

In the MRMDF architecture, the input sequence and the output sequence are in the specified order. The order of the output sequence is the bit reversal of the order of the input sequence, as seen in Fig. 4. The operation of the FFT or IFFT is controlled by the control signal, FFT/IFFT, as shown in Fig. 4. When an IFFT is performed in our processor, the sign of the imaginary part of the input sequences will be changed and then they will be performed by the process in treating FFT. The sign of the imaginary part of output data from FFT will be changed again and then will be divided by 128. Because 128 is a power of two, the operation of the division is implemented by shifting the decimal point location. The function of Module 1 is to implement a radix-2 FFT algorithm, corresponding to the first stage of the SFG, as shown in Fig. 3. Modules 2 and 3 are to realize the radix-8 FFT algorithm, corresponding to the second and third stages of the SFG, as displayed in Fig. 3. In order to minimize the memory requirement and to ensure the correction of the FFT output data, two different structures are built in Modules 2 and 3 to implement the radix-8 FFT algorithm. In addition, the hardware complexity of the complex multiplier will be also considered in the proposed architecture. In the next few subsections, each Module will be described in more detail.

1) Module 1: Module 1 consists of a register file which can store 64 pieces of complex data, one butterfly unit (BU), two complex multipliers, two ROMs, and some multiplexers, as shown in Fig. 5. The function of ROM is used to store twiddle factors. Only $1/8$ period of cosine and sine waveforms are stored in ROM, and the other period waveforms can be reconstructed by these stored values. The BU consists of four BU_2s, which operate the complex addition and complex subtraction from two input data. Because the radix-2 FFT algorithm is adopted in this module, BU cannot start until both the input sequences $x(n)$ and $x(64+n)$ are available. This corresponds to the first stage of SFG, as shown in Fig. 3. The order of the four parallel input sequences in Module 1 is $\text{in}(4m), \text{in}(4m+1), \text{in}(4m+2)$, and $\text{in}(4m+3)$, respectively, where m is from $0 \ldots 31$. Therefore, these two available data of each data path are separated by 16 cycles if one input data of each path is available per clock cycle. At the first 16 cycles, the first 64 data are stored in the register file. At the next 16 cycles, the eight input data $x(i)$ and $y(i)$ of the BU are received from the register file and the input, respectively, as shown in Fig. 5. Then the BU generates the outputs data according to the radix-2 FFT algorithm. Meanwhile, four output data, $X(i)$, generated by the BU, are fed to Module 2 directly, and the other four output data $Y(i)$ are stored into the register file. After 32 cycles, these data $Y(i)$ are read from the register file and are multiplied by the twiddle factors simultaneously before they are sent to Module 2. In general, four complex multipliers are needed in the four-parallel approach to implement the radix-2 FFT algorithm. Also, the utilization rate of the complex multiplier is only 50%. This paper proposes a new approach to increase the utilization rate and to reduce the number of complex multipliers. The detailed operation is described below. When the $Y(i)$'s are generated by the BU, two of the $Y(i)$'s, $Y(1)$ and $Y(2)$, are multiplied by the appropriate twiddle factors first before the $Y(i)$'s are stored in the register file. After 32 clock cycles, other two $Y(i)$'s, $Y(3)$, and $Y(4)$, are multiplied before the data $Y(i)$'s are fed to Module 2. By rescheduling the time of the complex multiplications, it is clear to find that only two complex multipliers are needed in our approach, as shown in Fig. 5. The utilization of the complex multipliers can achieve 100% by using our proposed approach.

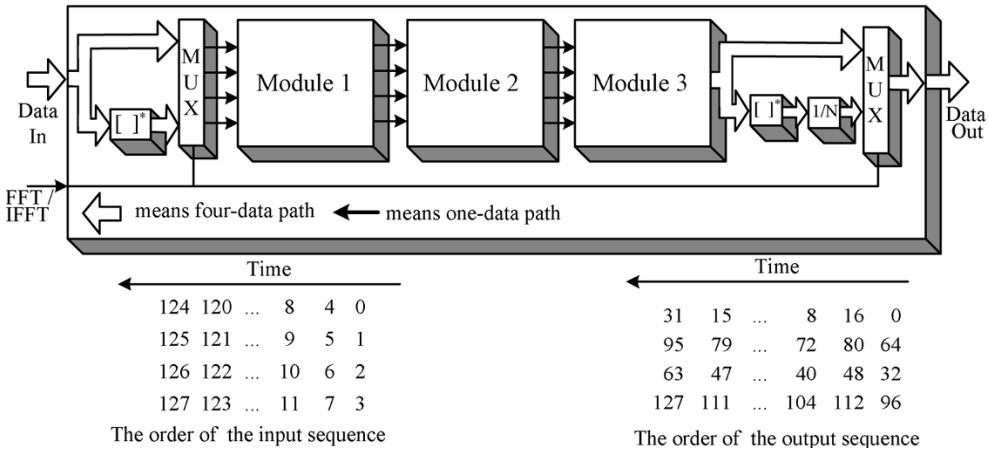


Fig. 4. Block diagram of the proposed 128-point FFT/IFFT processor.

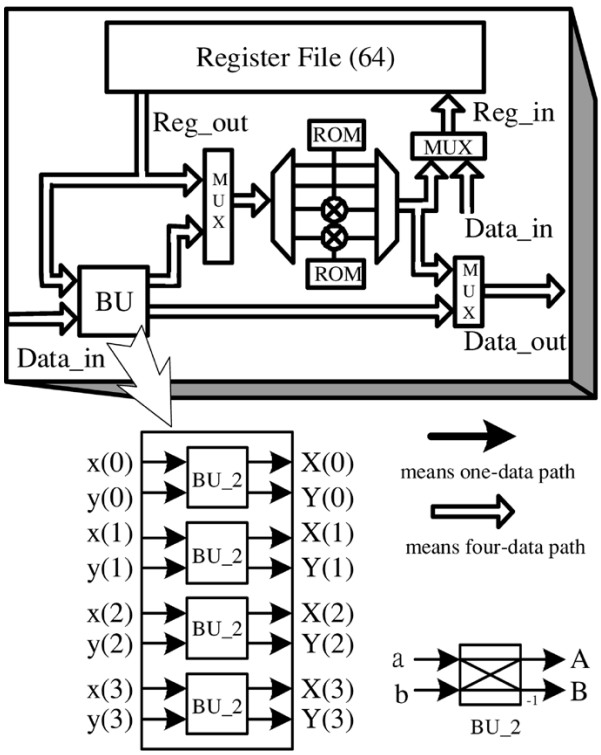


Fig. 5. Block diagram of the Module 1.

2) Module 2: Module 2 consists of four BU_8 structures and one modified complex multiplier, as shown in Fig. 6. These four BU_8s operate in the same way. The architecture of BU_8 is directly mapped from the three-step radix-8 FFT algorithm, whose SFG is shown in Fig. 3. Also, the sizes of the three delay elements in the BU_8 are eight, four, and two points, respectively, as shown in Fig. 6. The function of the delay element is to store the input data until the other available input data are received for the BU_2 operation. The output data generated by the BU_2 in the first step and second step are multiplied by a trivial twiddle factor, $1, -j, W_8^1$ or W_8^3 before they are fed to the next step. These twiddle factors can be implemented efficiently, as mentioned in Section II. However, the four output data from the third step of the BU_8 need to be multiplied by the nontrivial twiddle factors simultaneously in the modified complex multiplier.

It is inefficient to build four complex multipliers for multiplying different twiddle factors simultaneously. So we modified an approach proposed by Maharatna *et al.* [10] to simplify the complexity of the complex multipliers. The twiddle factors of the modified complex multiplier are $W_{64}^{p}(e^{(-j2\pi p)/(64)}) = X_p + jY_p$, where $X_p = \cos((2\pi p)/(64))$ and $Y_p = \sin((2\pi p)/(64))$ are the real and imaginary parts of the twiddle factor and p is from 0 to 49, as shown in Fig. 7. However, only nine sets of constant values, (X_p, Y_p) with p =0–8 in region A are needed, because the twiddle factor in the other seven regions can be obtained by using the mapping table, as seen in Table I. In practice, we only need to implement eight sets of constant values in the A region, since the first set of constant values (1, 0) is trivial. In addition, these constant values can be realized more efficiently by using several adders and shifters [10].

Table II shows the scheduling of the twiddle factor in each data path after the twiddle factors are mapped to region A. It can be clearly seen that the twiddle factor of four paths in each time slot has different values, except for time slots 2 and 3. In time slots 2 and 3, the hardware conflict will happen if only one constant multiplier 4 is built. Therefore, an additional constant multiplier 4 is used in our design to avoid spending one more cycle. The block diagram of the modified complex multiplier is shown in Fig. 8. In the beginning, the four output sequences from the third step of the BU_8 are separated into real and imaginary parts. The data of each path are fed to appropriate constant multipliers according to the scheduling of the twiddle factor, as shown in Table II. Therefore, the entire constant multiplication calculation can be implemented by just using eight sets of constant values by swapping the real and imaginary parts appropriately and choosing the appropriate sign, following the mapping table (Table I). The gate count of this approach can save about 38% compared to the four-complex-multiplier approach, and the performance of this approach is equivalent to that of the four complex multipliers.

3) Module 3: The radix-8 FFT algorithm is realized in Module 3. The detailed block diagram of Module 3 is shown in Fig. 9. The structure of Module 3 is different from that of Module 2, because the two available data of the BU_2 in the second and third steps are in different data paths. Thus, a

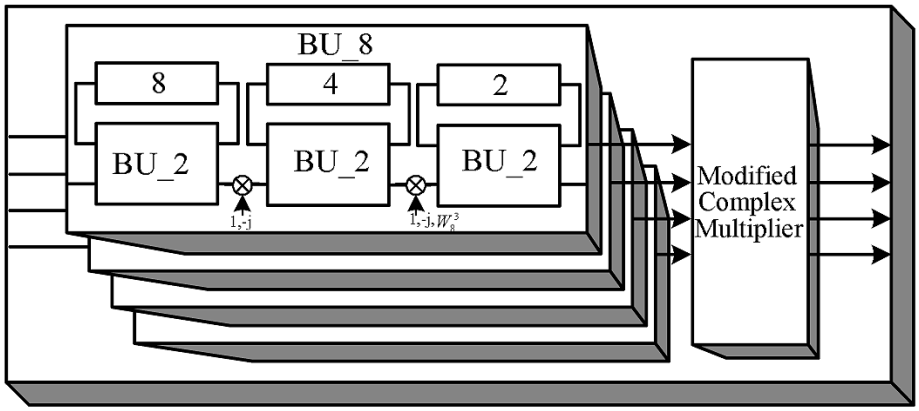


Fig. 6. Block diagram of the Module 2.

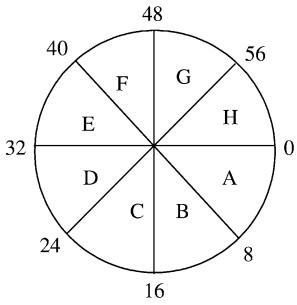


Fig. 7. Twiddle factors are divided into eight regions.

TABLE I
MAPPING TABLE USED TO DETERMINE THE VALUES OF THE TWIDDLE FACTORS IN DIFFERENT REGIONS

Region	Real	Imaginary
A	X_p	Y_p
B	$-Y_p$	$-X_p$
C	Y_p	$-X_p$
D	$-X_p$	Y_p
E	$-X_p$	$-Y_p$
F	Y_p	X_p
G	$-Y_p$	X_p
H	X_p	$-Y_p$

suitable structure is needed to ensure the correction of the FFT output data. Some output data, generated by the BU_2 in the first and second steps, are multiplied by the nontrivial twiddle factors before they are fed to the next step.

B. Comparison

In general, the performance and hardware cost of the pipelined FFT architecture are increased by using the multiple data-path approach. Thus, the multipath-based architecture usually provides higher throughput rate with higher hardware cost if the parallel input data can be supported in this approach.

The proposed MRMDF architecture hardware costs in terms of 128-point FFT are as follows:

- registers number: 124;
- complex multipliers: $2 + 4 \times 0.62$, where the complexity of modified complex multiplier is only 62% of that of four complex multipliers;
- complex adders: 48.

TABLE II
SCHEDULING OF THE TWIDDLE FACTOR, W_{64}^{p}, WHERE p IS FROM 1 TO 8

Time slot / Data_path	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1 st	0	0	0	0	0	8	0	8	0	4	0	4	0	4	0	4
2 nd	0	0	4	4	2	6	6	2	1	5	5	7	3	1	7	3
3 rd	0	0	8	8	4	4	4	4	2	6	6	2	6	2	2	6
4 th	0	0	4	4	6	2	2	6	3	7	1	3	7	5	5	1

Table III compares the hardware requirement, FFT algorithm, and throughput rate with several classical and proposed approaches in the 128-point FFT. The known MDC architectures like R4MDC [6] and the architecture proposed by Jung *et al.* [11] are not suitable for the 128-point FFT in UWB applications, because the FFT size used in their approaches is limited by a power of 4. In order that these two architectures are able to process the 128-point FFT, we modify both architectures by adding the proposed Module 1 to them. In addition, the throughput rate of the traditional MDC architecture is raised by increasing the utilization of butterfly units; this can be done by reordering the appropriate parallel input data in the input buffer before the data are loaded into the FFT processor. Consequently, the revised R4MDC and Jung's architectures can be compared with ours, as seen in Table III. It should be emphasized that the input buffer, whose size is usually proportional to the number of data paths, is needed in all FFT processors listed in Table III, except for our proposed MRMDF architecture and R2^{3}SDF architecture. By combining the features of the R2^{3}SDF and the R4MDC approaches, the proposed FFT architecture not only can implement the radix-8 FFT algorithm in a 128-point FFT to reduce the number of complex multiplications but also can provide four times the throughput rate, compared with the R2^{3}SDF scheme, as listed in Table III. In addition, the numbers of register excluding the input buffer and complex multiplier used in our scheme are only 38.9% and 44.8% of those in the SRMDC architecture [7]. Although the number of complex adders in our design is greater than that in the others, the cost of

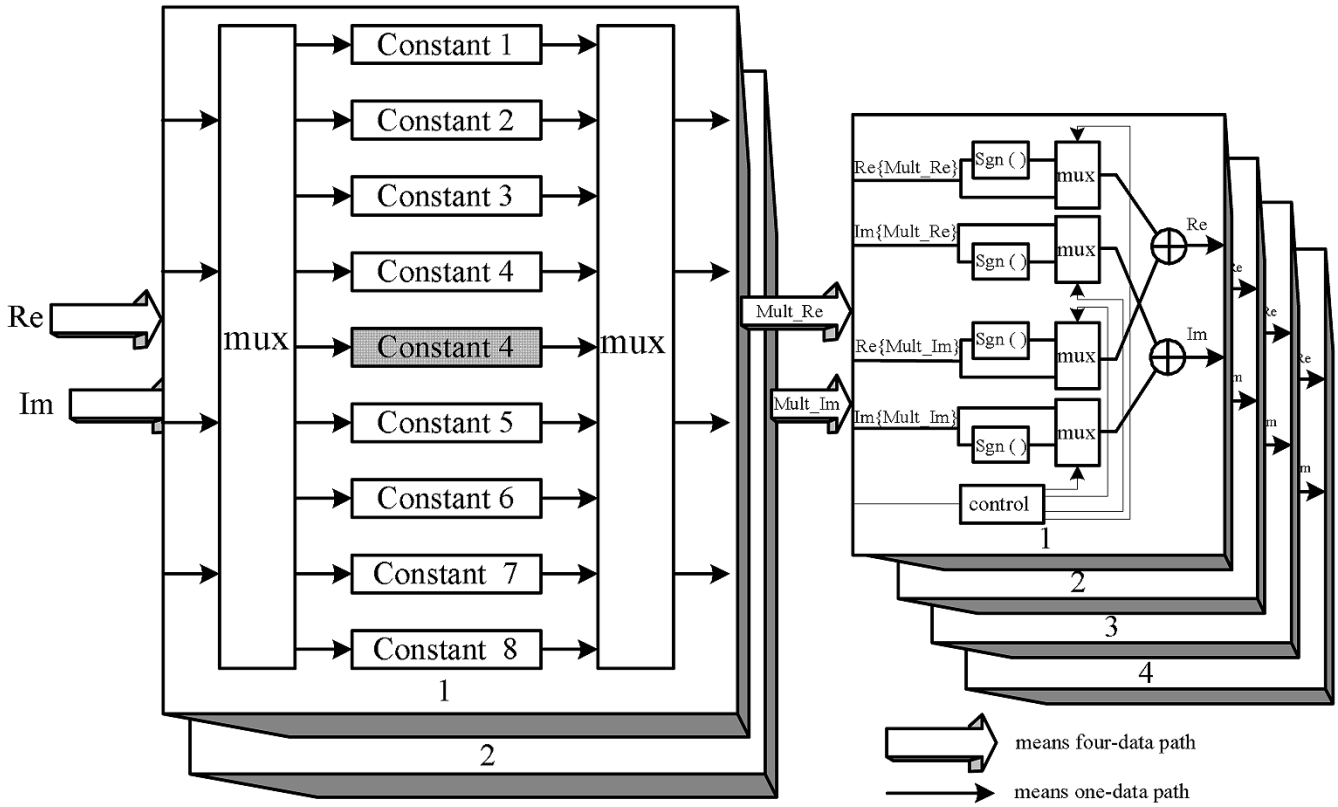


Fig. 8. Block diagram of the modified complex multiplier.

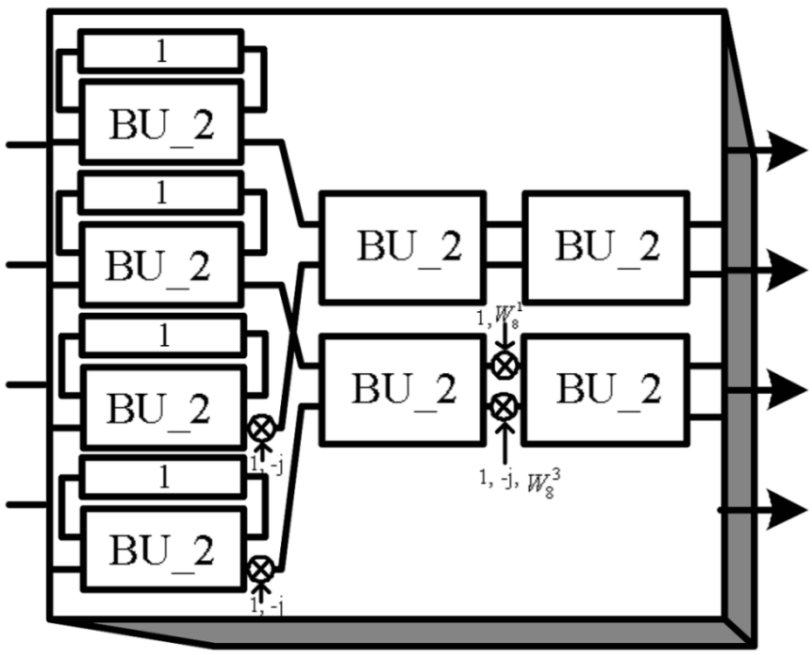


Fig. 9. Block diagram of Module 3.

complex adders is much less than that of registers and complex multipliers, respectively.

V. Simulation

The appropriate word length in an FFT/IFFT processor is determined by the fixed-point simulation before hardware implementation. Fig. 10 shows the simulation results for the relation of SNR with the internal word length of the FFT/IFFT processor. In Fig. 10, IN SNR means the ratio of the signal to the AWGN coming from the channel at the FFT input; OUT SNR means the ratio of the signal to the AWGN plus the quantization noise generated by the operation of the finite word and truncation in a FFT/IFFT processor at the output FFT. As seen in the figure,

TABLE III
COMPARISON OF THE 128-POINT PIPELINED FFT ARCHITECTURE

	MRMDF (proposed)	SRMDC*[7]	R2MDC*[6]	R2^{3}SDF [3]	Modified R4MDC* [6]	Modified Yunho Jung et al.* [11]
No. of registers (complex words)	124 (38.9%)	318 (100%)	190 (60%)	127 (40%)	220 (69%)	220 (69%)
No. of complex multipliers	2+4x0.62 (44.8%)	10 (100%)	6 (60%)	3 (30%)	8 (80%)	6 (60 %)
No. of complex adders	48 (100%)	34 (70.8%)	14 (29%)	14 (29%)	26 (54%)	26 (54%)
Algorithm	Radix-2 Radix-8	Split-radix	Radix-2	Radix-2 Radix-8	Radix-2 Radix-8	Radix-2 Radix-8
Input data format	Parallel (4 input port)	Parallel (4 input port)	Parallel (2 input port)	Serial	Parallel (4 input port)	Parallel (4 input port)
Output data format	Parallel (4 output port)	Parallel (6 output port)	Parallel (2 output port)	Serial	Parallel (4 output port)	Parallel (4 output port)
Throughput rate (R : clock rate)	4R	$6R\times0.73$	2R	R	4R	4R

*(1) No. of registers excluding the input buffer is listed. (2) The desired throughput rate can be achieved if employing appropriately reordered parallel input data.

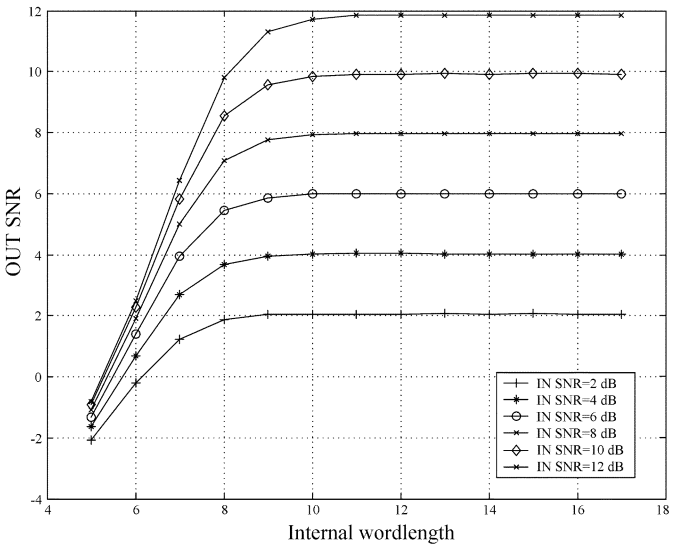


Fig. 10. Simulation result for SNR versus the internal word length.

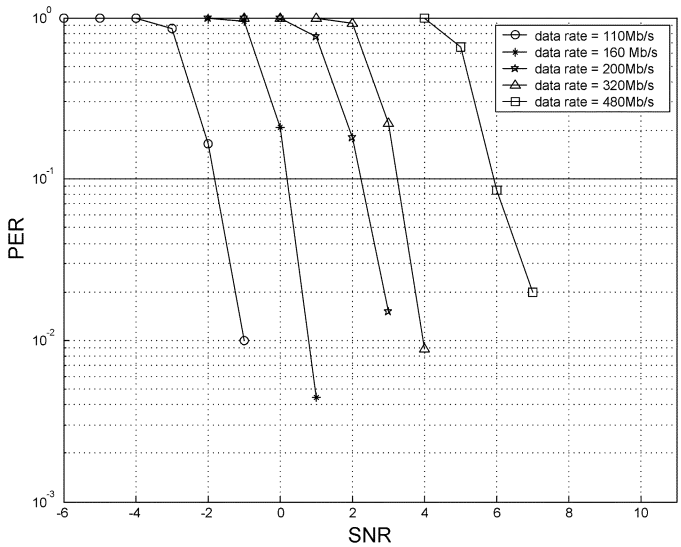


Fig. 11. Performance analysis in a complete UWB system.

keeping IN SNR fixed, the OUT SNR increases with increasing word length by suppressing the quantization noise, but it tends to saturate when the OUT SNR equals to the IN SNR. This means that, as the OUT SNR reaches saturation, the quantization noise can be almost ignored. Based on the simulation results, we determined the word length of the proposed FFT/IFFT to be 10 in both the real and imaginary parts. The determined word length not only keeps the quantization noise to the least value but also can minimize the hardware complexity.

With the chosen word length of the proposed FFT/IFFT processor, system performance is evaluated through the complete UWB system platform. We consider the cases of different data rates (110, 160, 200, 320, and 480 Mb/s) with code rates $11/32, 5/8$, and $3/4$ in the AWGN channel. The package error rate (PER) plotted against SNR is shown in Fig. 11. From the figure, it is clear to see that the proposed FFT processor can provide a satisfactory coded receiver performance (PER below 0.08) under the AWGN channel.

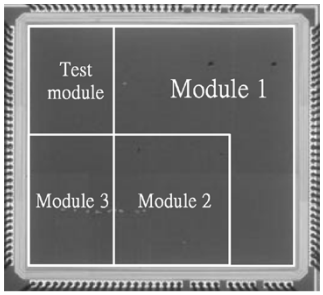


Fig. 12. Microphotograph of the proposed FFT/IFFT processor.

TABLE IV
CHIP SUMMARY OF THE PROPOSED FFT/IFFT PROCESSOR

Items	Specification
FFT size	*128 point*
Process Technology	*0.18 μm 1p6m*
Package	*144 CQFP*
Supply Voltage	*3.3 V/1.8V*
Max Work Frequency	*250 MHz*
Throughput rate	*1G sample/s*
Core Power Consumption	*175 mW @ 250 MHz* *77.6 mW @ 110 MHz*

VI. CHIP IMPLEMENTATION

After the appropriate word length of the proposed FFT/IFFT processor is chosen, the architecture of the processor was modeled in Verilog and functionally verified using Verilog-XL simulator. The output data from the Verilog coded architecture agreed with the output data of the FFT/IFFT in our UWB platform, which is coded by MATLAB. This test chip of the 128-point FFT/IFFT processor is fabricated in 0.18-μm one-poly six-metal (1P6M) CMOS process. The core size including the test module is 1.76×1.76 mm^2, as shown in Fig. 12. The function of the test module consisting of 3.072-kb SRAM is used to save 24 chip pins. Input data are stored serially in the test module from the chip input pins before the operation of the processor. The test module provides four complex data in parallel to the FFT/IFFT processor core when the processor begins to work. The 140-pin chip is packaged in 144-pin CQFP package, where 78 pins are signal pins and others are power pins.

Table IV lists the chip summary and measurement results. The highest throughput rate of our proposed architecture is up to 1 Gsample/s with power dissipation of 175 mW at 250 MHz. According to the specifications of UWB system, the throughput rate of the FFT/IFFT is 409.6 Msample/s. At the working frequency of 110 MHz, the power consumption of the FFT/IFFT processor is only 77.6 mW for 480 Mbs/s.

VII. CONCLUSION

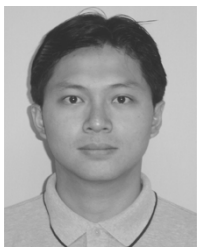
In this paper, a novel 128-point FFT/IFFT processor for OFDM-based UWB systems has been proposed. In our proposed MRMDF architecture, high throughput rate can be achieved by using four data paths. Furthermore, the hardware costs of memory and complex multiplier can be saved by adopting delay feedback and data scheduling approaches. In addition, the number of complex multiplications is reduced effectively by using a higher radix algorithm. This test chip has been designed, fabricated, and tested in 0.18-μm CMOS process. The measurement results show that the throughput rate of this test chip is up to 1 Gsample/s while it dissipates 175 mW. When the throughput rate of our processor meets UWB standard, it only consumes 77.6 mW.

ACKNOWLEDGMENT

The authors would like to thank their colleagues of the SI2 group of National Chiao Tung University for many fruitful discussions.

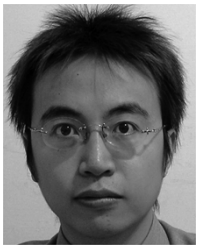
REFERENCES

[1] A. Batra *et al.*, "Multi-Band OFDM Physical Layer Proposal for IEEE 802.15 Task Group 3a," IEEE P802.15-03/268r3, Mar. 2004.
[2] 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.
[3] S. He and M. Torkelson, "Designing pipeline FFT processor for OFDM (de)modulation," in *Proc. URSI Int. Symp. Signals, Systems, and Electronics*, vol. 29, Oct. 1998, pp. 257–262.
[4] J. O'Brien, J. Mather, and B. Holland, "A 200 MIPS single-chip 1 k FFT processor," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, vol. 36, 1989, pp. 166–167.
[5] 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.
[6] L. R. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1975.
[7] J. García, J. A. Michel, and A. M. Burón, "VLSI configurable delay commutator for a pipeline split radix FFT architecture," *IEEE Trans. Signal Process.*, vol. 47, no. 11, pp. 3098–3107, Nov. 1999.
[8] 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.
[9] L. Jia, Y. Gao, J. Isoaho, and H. Tenhunen, "A new VLSI-oriented FFT algorithm and implement," in *Proc. 11th Annu. IEEE Int. ASIC Conf.*, Sep. 1998, pp. 337–341.
[10] K. Maharatna, E. Grass, and U. Jagdhold, "A 64-point fourier transform chip for high-speed wireless lan application using OFDM," *IEEE J. Solid-State Circuits*, vol. 39, no. 3, pp. 484–493, Mar. 2004.
[11] Y. Jung, H. Yoon, and J. Kim, "New efficient FFT algorithm and pipeline implementation results for OFDM/DMT applications," *IEEE Trans. Consum. Electron.*, vol. 49, no. 1, pp. 14–20, Feb. 2003.



Yu-Wei Lin was born in Tainan, Taiwan, R.O.C., in 1975. He received the B.S. degree in electrical engineering from National Sun Yat-Sen University, Kaohsiung, Taiwan, R.O.C., in 1999 and the M.S. degree in electronics engineering from National Chiao Tung University, Hsinchu, Taiwan, R.O.C., in 2003, where he is currently working toward the Ph.D. degree.

His research interests include baseband signal processing, VLSI architectures, and SoC designs for communication systems.



Hsuan-Yu Liu was born in Taipei, Taiwan, R.O.C., on May 9, 1977. He received the B.S. and M.S. degrees in electronics engineering from National Chiao Tung University, Hsinchu, Taiwan, R.O.C., in 1999 and 2001, respectively, where he is currently working toward the Ph.D. degree.

His research interests include VLSI architecture, low-power SoC, and wireless communication system, especially in OFDM-based baseband transceivers for high-speed WLANs and ultrawideband (UWB) systems.

Chen-Yi Lee (M'01) received the B.S. degree from National Chiao Tung University, Hsinchu, Taiwan, R.O.C., in 1982, and the M.S. and Ph.D. degrees from Katholieke University Leuven (KUL), Leuven, Belgium, in 1986 and 1990, respectively, all in electrical engineering.

From 1986 to 1990, he was with IMEC/VSDM, working in the area of architecture synthesis for digital signal processing (DSP). In February 1991, he joined the faculty of the Electronics Engineering Department, National Chiao Tung University, where he is currently a Professor and Department Chair. His research interests mainly include VLSI algorithms and architectures for high-throughput DSP applications. He is also active in various aspects of high-speed networking, system-on-chip design technology, very low power designs, and multimedia signal processing. He served as the Director of Chip Implementation Center (CIC), an organization for IC design promotion in Taiwan. He is now the Microelectronics Program Coordinator of the Engineering Division under National Science Council of Taiwan.

Prof. Lee was the former IEEE Circuits and Systems Society Taipei Chapter Chair.