EURASIP Journal on Applied Signal Processing 2003:13, 1306–1316

VLSI Design of a Variable-Length FFT/IFFT Processor for OFDM-Based Communication Systems

Jen-Chih Kuo
Graduate Institute of Electronics Engineering and Department of Electrical Engineering, National Taiwan University, Taipei 106, Taiwan
Email: jj@access.ee.ntu.edu.tw

Ching-Hua Wen
Graduate Institute of Electronics Engineering and Department of Electrical Engineering, National Taiwan University, Taipei 106, Taiwan
Email: johnny@access.ee.ntu.edu.tw

Chih-Hsiu Lin
Graduate Institute of Electronics Engineering and Department of Electrical Engineering, National Taiwan University, Taipei 106, Taiwan
Email: fil@access.ee.ntu.edu.tw

An-Yeu (Andy) Wu
Graduate Institute of Electronics Engineering and Department of Electrical Engineering, National Taiwan University, Taipei 106, Taiwan
Email: andywu@cc.ee.ntu.edu.tw

Received 30 January 2003 and in revised form 10 July 2003

The technique of orthogonal frequency division multiplexing (OFDM) is famous for its robustness against frequency-selective fading channel. This technique has been widely used in many wired and wireless communication systems. In general, the fast Fourier transform (FFT) and inverse FFT (IFFT) operations are used as the modulation/demodulation kernel in the OFDM systems, and the sizes of FFT/IFFT operations are varied in different applications of OFDM systems. In this paper, we design and implement a variable-length prototype FFT/IFFT processor to cover different specifications of OFDM applications. The cached-memory FFT architecture is our suggested VLSI system architecture to design the prototype FFT/IFFT processor for the consideration of low-power consumption. We also implement the twiddle factor butterfly processing element (PE) based on the coordinate rotation digital computer (CORDIC) algorithm, which avoids the use of conventional multiplication-and-accumulation unit, but evaluates the trigonometric functions using only add-and-shift operations. Finally, we implement a variable-length prototype FFT/IFFT processor with TSMC 0.35 μm 1P4M CMOS technology. The simulations results show that the chip can perform (64–2048)-point FFT/IFFT operations up to 80 MHz operating frequency which can meet the speed requirement of most OFDM standards such as WLAN, ADSL, VDSL ($256 \sim 2K$), DAB, and $2k$-mode DVB.

Keywords and phrases: cached FFT, mixed-scaling and rotation CORDIC, and OFDM communications.

1. INTRODUCTION

The orthogonal frequency division multiplexing (OFDM) system is a form of multicarrier modulation (MCM) technologies [1, 2, 3]. Due to its robustness against frequency-selective fading or narrowband interference, the OFDM technology has been widely implemented in many digital communications such as wireless local area network (WLAN, IEEE 801.11a/g), digital audio/video broadcasting (DAB/DVB), asymmetric DSL (ADSL), and very-high-speed DSL (VDSL) systems [4, 5, 6]. The most important modulation/demodulation kernel in OFDM system is the fast Fourier transform and inverse FFT (FFT/IFFT) operations. However, the size and execute time, T_{FFT}, of the FFT/IFFT processors is different for various applications of OFDM systems, as shown in Table 1 [7, 8]. For example, the FFT/IFFT processor must execute 64 points in 3.2 microseconds for the WLAN system, and execute 2048-point operation in

Table 1: FFT/IFFT size for OFDM-based communication systems.

Application	FFT/IFFT size (points)	T_{FFT} (μs)
WLAN	64	3.2
ADSL	2×256	231
VDSL	$2 \times 256 \times 2^n$, $n = 0, \ldots, 4$	231
DAB	256×2^n, $n = 0, \ldots, 3$	31×2^n
DVB-T	8192/2048	896/224

224 microseconds for the DAB system. With the prevalence of multimode/multistandard communication systems, it would be desirable to have a various points of FFT/IFFT processor to fit most OFDM systems.

In this paper, motivated by this trend, we design and implement a variable-length FFT/IFFT processor that can fit various existing OFDM-based communication systems. Firstly, we adopt the newly proposed cached-memory FFT architecture [9] as our suggested VLSI system architecture to lower the main memory access times to reduce the memory power consumption. In addition, we employ mixed-scaling rotation CORDIC (MSR-CORDIC) algorithm [10] to design the butterfly processing element (PE). It can overcome the problems of the long iteration number and large quantization noise encountered in conventional CORDIC-based PE designs. Besides, this design approach can help to reduce the switching activities, hence to achieve low-power consumption in portable communication applications. Finally, we design and implement the variable-length prototype FFT/IFFT processor with TSMC 0.35 μm 1P4M CMOS technology. The die area of the FFT/IFFT processor is 15.21 mm^2 including 2048$\times$32 bits SRAM. The maximum operating frequency can be up to 80 MHz, which can meet most existing OFDM systems using (64–2048)-point FFT/IFFT operations.

The rest of this paper is organized as follows. Section 2 reviews the cached FFT algorithm. We also present our proposed system architecture in this section. Section 3 discusses the MSR-CORDIC-based PE design. Section 4 discusses the design issues of the address generator (AG) and control logic unit (CLU), respectively. Then, in Section 5, we estimate the word length of the FFT hardware architecture. The VLSI implementation results and comparisons are made in Section 6. Finally, we conclude this work in Section 7.

2. CACHED FFT ARCHITECTURE

2.1. Review of cached FFT architecture

There are various structures for the implementations of FFT processor, such as single memory, dual memory, pipelined architecture, and array type [9]. Typically, conventional FFT algorithms are developed to minimize the number of multiplications and additions while maintaining a simple form. However, the hidden memory operations are usually ignored. The hidden memory operations might take half of the power consumption of the whole FFT calculation [11]. To reduce the number of memory access, we adopt the cached-memory architecture [9] to realize the proposed variable-length FFT processor. The cached FFT is designed explicitly to operate on a processor with a hierarchical memory system. By taking advantage of a small cache memory, the algorithm enables higher operating clock frequencies and reduces data communication energy. On the other hand, most of the memory can operate at lower frequency with only a small portion operates at higher speed. This could reduce the power dissipation.

Figure 1 shows the traditional 64-point FFT dataflow diagram. We can see that the whole data are read and written back to the main memory in every stage. As the FFT size increases, stage number will be increased. As a result, the data movement will cause much power consumption and long execution time.

The basic idea of cached FFT is to reduce the number of main memory access as shown in Figure 2. Instead of processing one stage of butterfly operation at a time, we store data in local storage and process more data in one *super-stage (Pass0, Pass1, Pass2)* at a period time. To achieve this, two sections of data movement operations differing from traditional FFT are designed as shown in Figure 3. The resulting operations are still very regular and will not increase much complexity. The data will read/write from the cache memory of each super stage. Super stage can greatly reduce the number of main memory access as the FFT size N becomes bigger.

The nomenclature of the cached FFT is defined by following stages [9].

(1) *Stage.* A stage is the part of an FFT where all N memory locations are read, processed by a butterfly, and written back once.
(2) *Super stage.* A super stage is the portion of the cached FFT algorithm where all N-data word are loaded into a cache, processed, and written back to the main memory once, which can be regarded as a super stage.
(3) *Group.* A group is the portion of a super stage where a block of data is read from the main memory into a cached, processed, and written back to the main memory.
(4) *Pass.* A pass is the portion of a group where each word in the cache is read, processed with a butterfly, and written back to the cache once.

The detailed operations of the cached FFT are as follows.

(1) Input data are loaded into an N-word main memory.
(2) The number C of the N words is assembled to a group. The C words are loaded into the cache memory.
(3) The data in the cache are being processed by PE and written back to cache. Repeat this step when all passes in a super stage is processed.
(4) Processed data in the cache are flushed to main memory.
(5) Steps 2, 3, and 4 are repeated until all N words have been processed once in a super stage.
(6) Steps 2, 3, 4, and 5 are repeated till all super stages are finished. Then the operation of the FFT is completed.

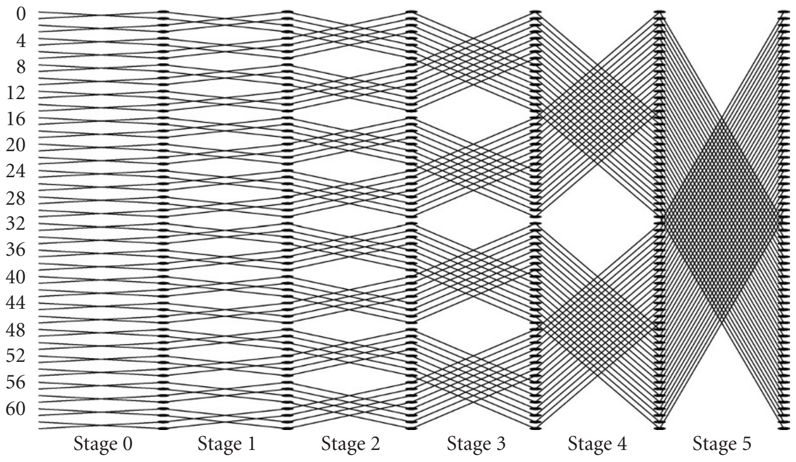


Figure 1: Traditional FFT dataflow diagram.

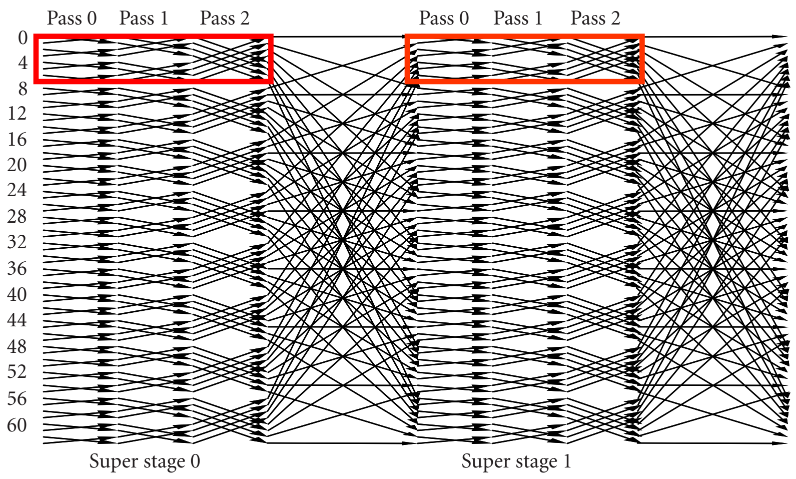


Figure 2: Cached FFT dataflow diagram [9].

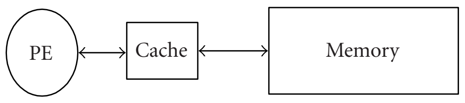


Figure 3: Cache-memory FFT processor architecture [9].

2.2. *Proposed system architecture of the variable-length FFT processor*

Based on the design concept of the cached FFT, we propose a variable-length FFT/IFFT processor architecture for OFDM application communication systems. It consists of three design units: CLU, AG, and PE, as shown in Figure 4. The CLU decides the forward/inverse FFT operation and the number of points that FFT will perform, based on the control word. The AG unit can generate the address that RAM and ROM accesses need. With the address, the coefficient ROM can output the twiddle factor to butterfly processing element. Then the PE can perform the complex arithmetic operations for the data of RAM and the coefficients stored in ROM. The RAM will store the input data as well as the temporary computed data and also play the role of the cache. The outputs of the FFT/IFFT results are also stored in the RAM for access. The following sections will discuss the design issues of those major functional units.

3. PROCESSING ELEMENT (PE) DESIGN

The coordinate rotational digital computer (CORDIC) algorithm is a well-known VLSI arithmetic unit. The basic concept of CORDIC is to decompose the desired rotation angles into several easy-to-be-implemented subangles [12, 13, 14]. The subangles can be implemented by the shift-and-add. In the traditional FFT hardware implementation, it needs many multipliers and adders to achieve the complex multiplications. This will always cause very large hardware requirement. In order to reduce the hardware complexity, many FFT

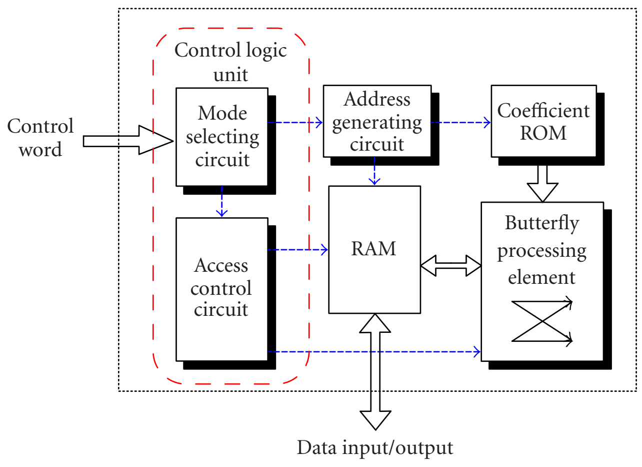


FIGURE 4: The proposed variable-length FFT/IFFT processor architecture.

processors have employed the CORDIC to perform twiddle factor operations. In this work, we adopted the newly proposed MSR-CORDIC scheme [10] to compose these subangles. With these easy-to-be-implemented subangles, the rotation operations can be performed easily. Besides, the hardware requirement of CORDIC is very simple. It also has potential advantage of low switching activity for low-power operations.

3.1. Review of MSR-CORDIC algorithm

In the conventional CORDIC algorithm [12], the scaling factor is always greater than 1. Therefore, it is necessary to scale down the norm of the input vector to its initial value after the rotation mode is finished. Furthermore, the signal-to-quantization-noise ratio (SQNR) will be reduced due to the growth of the scaling factor. To avoid the overhead of the scaling operation, the product of the scaling factors must be equal to 1. To overcome these problems, the range of the scaling factors must be greater and less than 1. The MSR-CORDIC algorithm [10] reformulates the iterative arithmetic as in Algorithm 1, where $\mu_i, \mu_j \in \{-1, 0, 1\}$; I and J denote the number of SPT terms of $x(n)$ and $y(n)$, respectively, and they are referred to as the *extending factors*; θ_n is the elementary angle and the initial value; $\overline{p_{n+1}}$ denotes the product of the scaling factors in nth iteration. The initial value of $\overline{p_1}$ is 1; N_{spt} are denoted as the number of SPT terms used in performing (1), which is the sum of I and J; N denotes the total number of iteration; $s_n \in \{0, 1, \dots, S\}$ and S denotes the number of maximum shift.

In the conventional CORDIC [12] and EEAS-CORDIC [15] algorithm, the norms of both schemes are enlarged after the micro-rotation operations. That is, the norm is amplified as illustrated in Figures 5a and 5b and Rm denotes the total iteration number of CORDIC. On the contrary, in MSR-CORDIC algorithm, (4) shows that the factor P_n can be either greater or less than 1. In Figures 5c and 5d, we depict the reachable points in 2D plane to emphasize the feature of P_n.

```
For $n = 0, 1, \dots, N$
I. *Rotation phase*
    $$\begin{bmatrix} x(n+1) \\ y(n+1) \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{J} \mu_j 2^{-s_j} & -\sum_{i=1}^{I} \mu_i 2^{-s_i} \\ \sum_{i=1}^{I} \mu_i 2^{-s_i} & \sum_{j=1}^{J} \mu_j 2^{-s_j} \end{bmatrix} \begin{bmatrix} x(n) \\ y(n) \end{bmatrix}. \quad (1)$$
    • Elementary angle
    $$\theta_{n+1} = \tan^{-1}\left(\frac{\sum_{i=1}^{I} \mu_i 2^{-s_i}}{\sum_{j=1}^{J} \mu_j 2^{-s_j}}\right). \quad (2)$$
    • Accumulation angle
    $$z(n+1) = z(n) + \theta_{n+1}. \quad (3)$$
II. *Scaling phase*
    • Scaling factor
    $$p_{n+1} = \sqrt{\left(\sum_{i=1}^{I} \mu_i 2^{-s_i}\right)^2 + \left(\sum_{j=1}^{J} \mu_j 2^{-s_j}\right)^2}. \quad (4)$$
    • Product of the scaling factor
    $$\overline{p_{n+1}} = \overline{p_n} \times p_{n+1}, \quad (5)$$
*End*
```

ALGORITHM 1

From Figure 5, some other interesting features of the proposed scheme are discussed below.

(1) According to (2), the angles in MSR-CORDIC is much denser than the conventional CORDIC and EEAS-CORDIC, hence, the MSR-CORDIC can reach the target angle with fewer iteration as shown in Figure 5d. Furthermore, if we design the parameters, *si*, *μi*, appropriately so that both the quantization error of rotation angles and norms meet the system performance requirement at the same time, then the scaling operation can be avoided. Since we do not need the extra scaling operations, the MSR-CORDIC is faster in computational speed and the corresponding hardware cost is reduced.

(2) In some applications, the rotation angles are larger than $\pi/4$, such as the twiddle factors in FFT. It is difficult for the conventional CORDIC to perform such a rotation angle. In MVR-CORDIC [16], the authors utilize the prerotation strategy to overcome the problem and have the improvement of error performance. However, extra hardware costs and also the computing speed decrease. On the contrary, in the newly proposed MSR-CORDIC algorithm, the reachable angles are distributed from 0 to 2π.

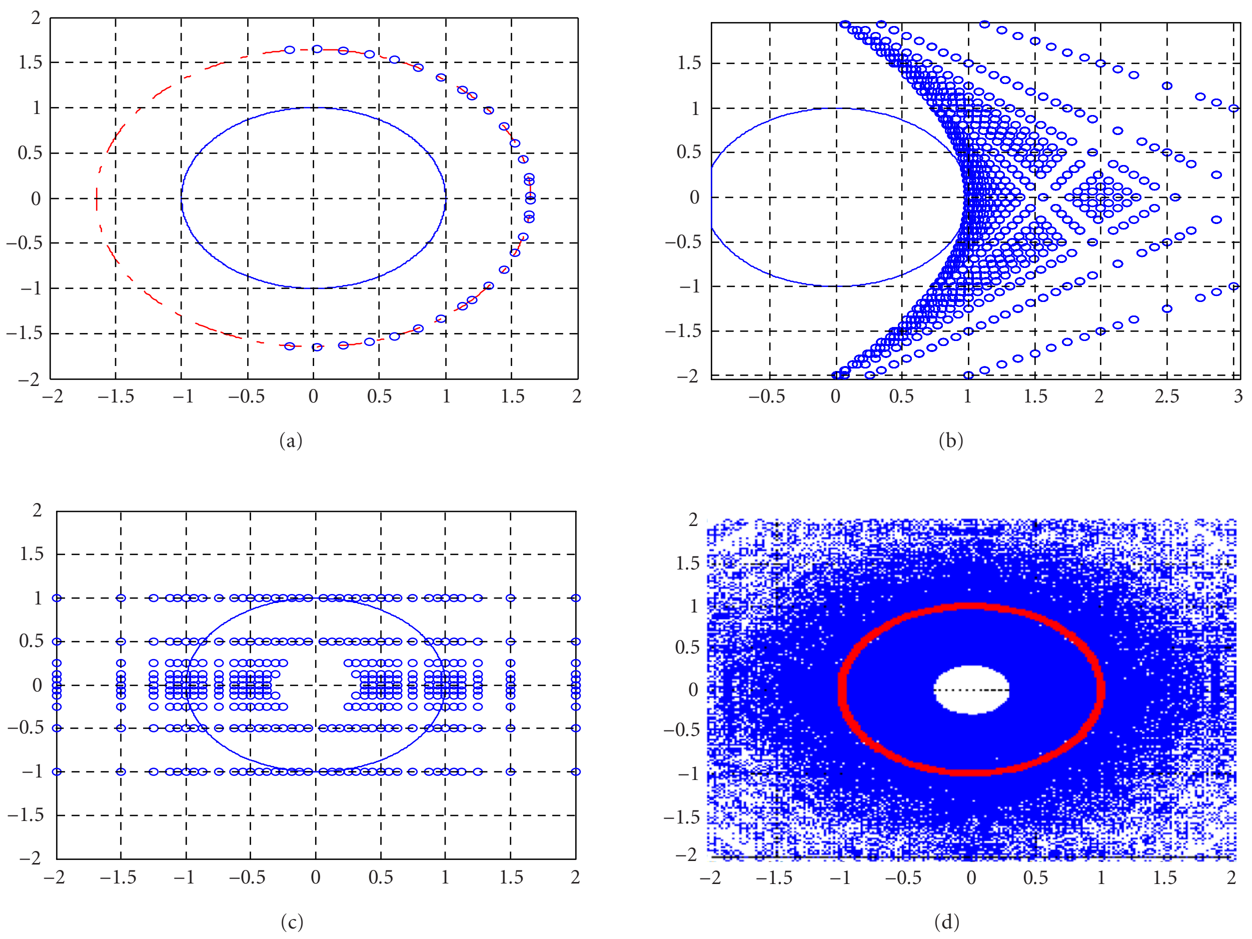


FIGURE 5: Constellation of reachable points under the rotation process. (a) Conventional CORDIC with $N = \text{Rm} = 4$. (b) EEAS-CORDIC with maximum shift range $S = 4$ and $\text{Rm} = 2$. (c) MSR-CORDIC with $I = 2$, $J = 1$, and $N = 1$. (d) MSR-CORDIC with $I = 2$, $J = 1$, and $N = 2$ for $1/3 \leqq Pn \leqq 3$ with maximum shift range $S = 4$.

3.2. *VLSI architecture of MSR-CORDIC for twiddle factor operation*

To realize the twiddle factor PE, we use the MSR-CORDIC-based VLSI architecture as shown in Figure 6. In order to enhance the speed, we arrange two-level pipeline in CORDIC design. The important differences from the conventional CORDIC design are the parameter sequence arrangement, physical consideration, and circuit speedup because of no-scaling operations. Based on (2), both of $x(n+1)$ and $y(n+1)$ are linear combination of their prior $x(n)$ and $y(n)$. All the coefficients of $x(n)$ and $y(n)$ are power of two numbers with the signs μ_i and μ_j, respectively. Hence, two Barrel shifter arrays (BSAs) are used to perform shifting operations. The number of the output signal is N_{spt} in each BSA. To perform the summation of the outputs, $2(N_{\text{spt}}-1)$ add/subtract operations must be performed and $2(N_{\text{spt}}-1)$ adders/subtractors are used or required to finish the process in one clock cycle. Therefore, in each MSR-CORDIC module, only $2(N_{\text{spt}} - 1)$ additions/subtractions are required.

3.3. *Low switching activity of PE unit*

The switching activity of coefficient may cause much power consumption in two's complement representation system. In CORDIC representation system, the representation of lower switching activity can be achieved. In Figure 7, we can see the different of switching activity between two's complement and CORDIC representations. The switching activity in CORDIC representation is lower than two's complement representation.

4. DESIGN OF ADDRESS GENERATOR UNIT AND CONTROL LOGIC UNIT

4.1. *Cached FFT/IFFT address generation [9]*

For a traditional *radix-r* N-point FFT, the stage number is $S = \log_r(N)$. By closely examining the data access pattern, we can recognize that butterflies are clustered into "Groups." We can represent the memory address by making use of two virtual counters:

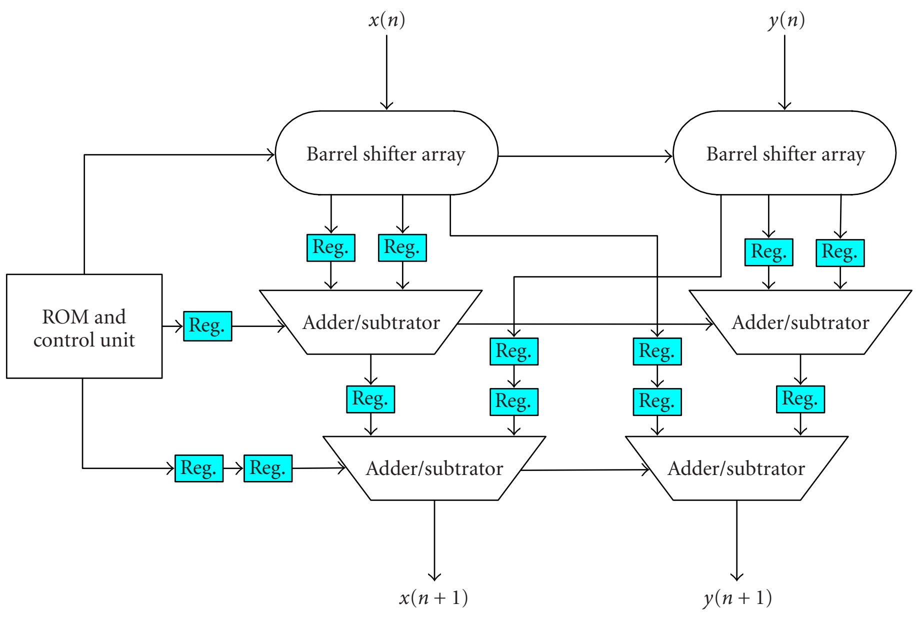


FIGURE 6: MSR-CORDIC-(Nspt = 3, I = 2, and J = 1) based PE structure with two-level pipeline.

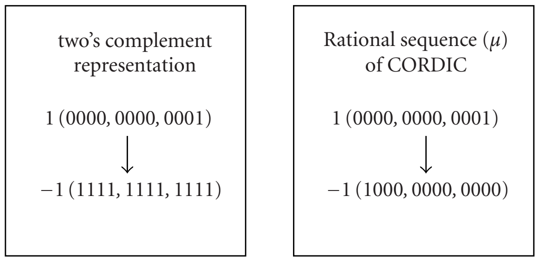


FIGURE 7: Switch activity in two's complement representation and CORDIC representation.

(1) *Group counter:* counts the groups of butterflies;
(2) *Butterfly counter:* counts butterfly in a group.

The cached FFT/IFFT address generation circuit can be viewed as a modified version of the traditional FFT. What we need to do is to find a grouping of the memory accesses such that a portion of the full FFT can be calculated using less than N words of memory. Fixing the number of bits in the group and butterfly counters, and keeping the positions of the group counter digits fixed across a super stage allows a subset of the FFT to be calculated in memory locations. The W_N coefficients are generated by using the same method that the traditional FFT uses, except that the new group and butterfly mappings are used.

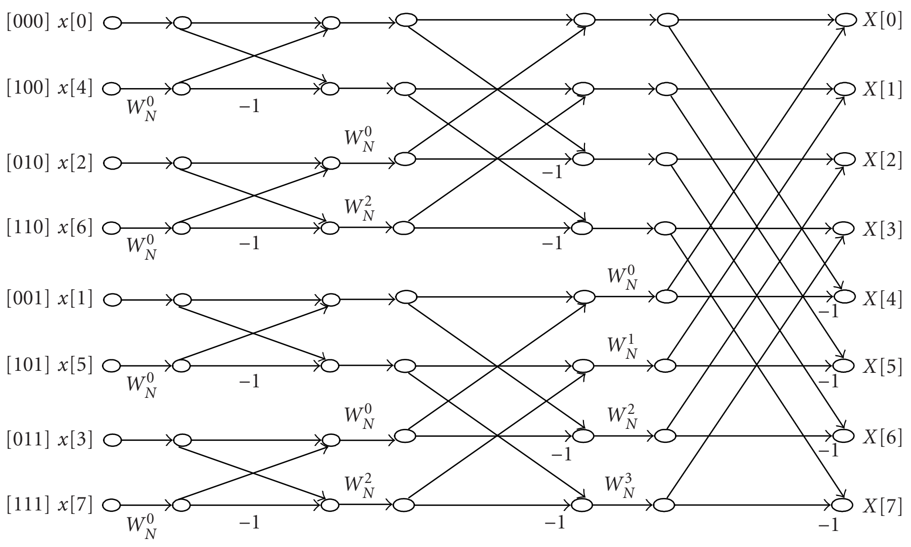
Similar to the traditional FFT case, we can map the above derivations into hardware realization by using an outer counter and inner counter. The former responds to the main memory address and the latter is to calculate the cache address. In any FFT transform, the length (N) and radix (r) must be specified. The cached FFT also requires the selection of number of super stage (E). So we define some terms, the size N of FFT and the number r of radix-r FFT; the number of passes per group $P = \log_r(N/E)$; the cache size $C = r^{\log_r N}/E = \sqrt[E]{N}$; the number of groups per super stage, $g = N/C$; the number of butterflies per pass $b = C/r$. Then the generalized form of memory and coefficient addresses for cached FFT [9] can be realized with some parameters. In order to perform (64–2048)-point FFT successfully, we choose the *radix* = 2. The radix-2 FFT can perform 64, 128, 256, 512, 1024, and 2048-point operations without any modification in the FFT hardware architecture. However, other radices FFT cannot perform all these points operations. For example, the radix-4 FFT can just perform 64, 256, and 1024-point FFT operations and cannot meet the specification of OFDM applications. Table 2 illustrates the address generated by AG for the 64-point radix-2 FFT.

4.2. Variable-length cached FFT/IFFT address generation

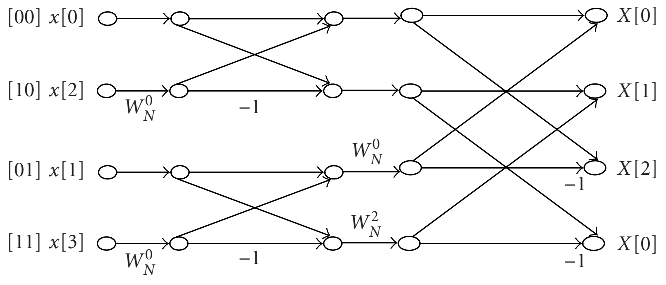
In order to achieve the address for the need of various point FFT operations, we will use the methodology of the address shared with the discarding operation. It means that the lower point FFT operations will discard some digits of the generated address and then share the rest of the generated address in the highest FFT operations. We just need to design the single address generator for the highest point FFT operations. For example, $x[0], x[1], \ldots, x[7]$ in Figure 8 is the data accessed from memory, and $X[0], X[1], \ldots, X[7]$ is the corresponding operated data after FFT operations. It is obvious that the hardware operating $x[0]$, $x[2]$, $x[4]$, and $x[6]$ in 8-point FFT is as the same as the hardware in 4-point FFT. If we discard the LSB digit in the generated address of the 8-point FFT processor in Figure 8a, the final address for the

Table 2: Memory and coefficient addresses for a 64-point, $r = 2$, and $E = 2$ cached FFT.

Super stage (E)	Pass	Memory address	Cache address	ROM address
1	0	$b_5\, b_4\, b_3\, b_2\, b_1\, b_0$	$b_2\, b_1\, b_0$	0 0 0 0 0
	1	$b_5\, b_4\, b_3\, b_2\, b_0\, b_1$	$b_2\, b_0\, b_1$	b_0 0 0 0 0
	2	$b_5\, b_4\, b_3\, b_0\, b_2\, b_1$	$b_0\, b_2\, b_1$	$b_1\, b_0$ 0 0 0
2	0	$b_2\, b_1\, b_0\, b_5\, b_4\, b_3$	$b_2\, b_1\, b_0$	$b_5\, b_4\, b_3$ 0 0
	1	$b_2\, b_0\, b_1\, b_5\, b_4\, b_3$	$b_2\, b_0\, b_1$	$b_0\, b_5\, b_4\, b_3$ 0
	2	$b_0\, b_2\, b_1\, b_5\, b_4\, b_3$	$b_0\, b_2\, b_1$	$b_1\, b_0\, b_5\, b_4\, b_3$



(a)



(b)

Figure 8: (a) The generated address in 8-point FFT. (b) The generated address after discarding the LSB in 4-point FFT.

4-point FFT in Figure 8b can be generated. In the variable-length FFT processor, one addresses that the generator of the 2048-point FFT is necessary. Then we will discard some LSB digits depending on the operating FFT size.

4.3. Control Logic Unit (CLU) design

The topmost CLU is composed of the following individual circuits.

(1) *FFT/IFFT operation selection:* there is a 1-bit input which determines that the FFT or the IFFT transform should be computed. This will affect the coefficient design and is described in Section 5.

(2) *FFT size selection:* 3-bit FFT size selecting inputs decides which length of the FFT should be calculated. The size options include 64, 128, 256, 512, 1024, and 2048-point operations.

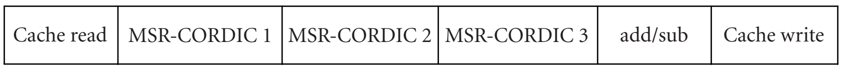


Figure 9: Six-stage cache-PE-cache pipelining scheme.

(3) *Data movement:* the processing kernel, PE, and cache main memory can operate at different frequency to further reduce the power consumption.

(4) *Pipeline of PE:* because the state of the FFT processor is independent of datum values, a deeply pipelined FFT processor is much less sensitive to pipeline hazards than general-purpose processor. Therefore, the clock speed and throughput rate can be dramatically increased with deeper pipelines. We have two-level pipeline, cache-PE-cache, and memory-cache-memory, as discussed below.

Cache-PE-cache pipeline

We have six-stage pipeline as shown in Figure 9. In the first stage, the input operands are read form the appropriate cache set and coefficient from ROM. Stages two to four are the three-stage CORDIC computation. Stage five will calculate the addition/subtract operation. The final stage will write back the result to the cache.

Memory-cache-memory pipeline

The cached FFT architecture significantly reduces the required movement of data to and from the main memory. The main memory arrays can be accessed at low frequency or with idle cycles. The cache is read in the first C stages and the data is written in the main memory. In the second C stages, the operation is the reverse operations of cache-memory pipeline.

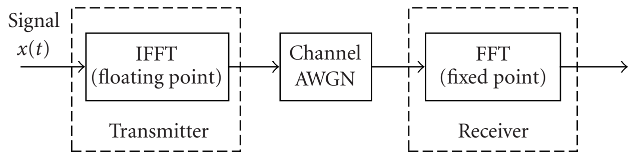


Figure 10: Fixed-point simulation environment.

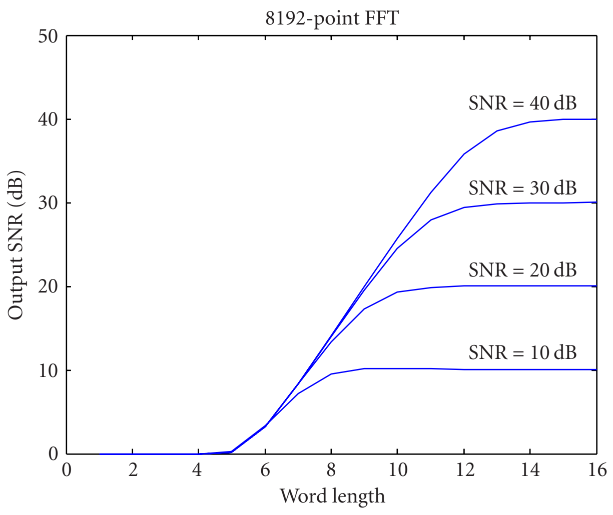


Figure 11: Fixed-point simulation of data word length.

5. FIXED-POINT SIMULATION

The FFT/IFFT processor is designed to perform various lengths of FFT/IFFT operations. For finite word-length definition, we have to do fixed-point simulation in the worst case. Based on the specification of OFDM system, the 8192-point FFT will suffer the largest quantization error as it has the most fixed-point calculation. Thus, we will perform system simulations to decide the finite word length of the hardware implementation with 8192-point FFT/IFFT operations. However, there are different channels and specifications for various OFDM applications and the channel models are usually not available. Therefore, we will consider the channel with only additive white Gaussian noise (AWGN). We will also refer to some academic and commercial works to choose parameter to decide the final word length. The simulation environment is shown in Figure 10. The transmitters are usually defined in the standards and have loose requirement than the receivers. Hence, we will only consider the FFT part in the receiver side. First, the signal $x(t)$ passes through the IFFT block and then through the channel with AWGN. The above operations have floating point precision. In the receiver side, the receiver signal is cut to B bits and performs the fixed-point FFT calculation.

We perform the system simulation of 8192-point FFT and then get the relationship between the output signal-to-noise ratio (SNR) and word length B, as shown in Figure 11. It is obvious that the FFT needs wider data word length when the signal-to-channel noise is better. That is because the quantization noise effect will become more important if the signal-to-channel noise is better. In the worst case of the signal to channel noise, 40 dBc, the 16-bit wide data word length can meet the requirement with 2-bit design margins. However, in the DVB-T receiver designs [17, 18, 19], the 16-bit wide data word length is adopted. Hence, we will choose the data word length as 16-bit wide. Figure 12 is the relationship between the coefficient word length and output SNR. The 12-bit coefficient word length could be a suitable implementation parameter. After referring to the designs [17, 18, 19], we choose the coefficient word length as 12-bit wide.

Table 3: Implementation results of the variable-length FFT processor.

Technology	TSMC 0.35 μm 1P4M CMOS
Voltage	3.3 V
Word length	16 bits
Gate counts	14, 732
Memory	2×1024 Word SRAM (32 bits)
Die size	$3.9 \times 3.9\,\text{mm}^2$
Core size	$2.6 \times 2.6\,\text{mm}^2$
Max frequency	80 MHz
Power range	126–574 mW
Power range per MHz	9.56–10.8 mW/MHz

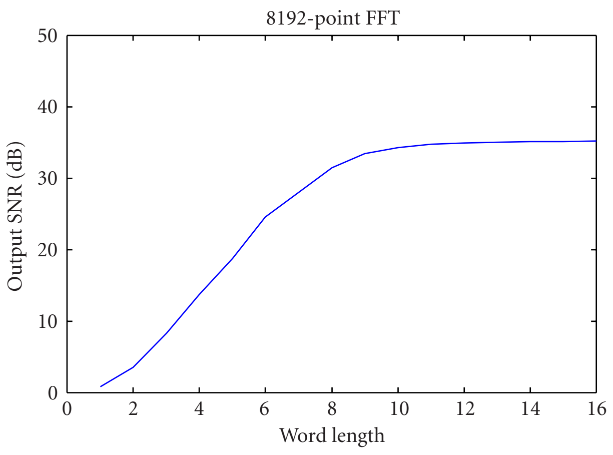


Figure 12: Fixed-point simulation of coefficient word length.

6. IMPLEMENTATION RESULT AND COMPARISON

The prototype FFT/IFFT processor is implemented with TSMC 0.35 μm 1P4M CMOS technology. The die area is $3.9 \times 3.9\,\text{mm}^2$ with 2048-word memory, each is 32 bits wide. The power normalized by the operating frequency ranges from 9.56 to 10.8 mW/MHz. The microphotograph of the processor is shown in Figure 13.

Table 3 lists the implementation result of this variable-length prototype FFT/IFFT. We can set the operating frequency that is based on the execute time of OFDM applications. For example, we have to set the operating frequency to 65 MHz if we want to apply it to the 802.11a WLAN system. The detailed information about the FFT sizes, the respective operating frequency, and the power consumption are listed in Table 4.

In order to eliminate the factor of different fabrication technology, we proposed the normalize index about the chip area and the power consumption. The first normalized index is a normalized area per FFT. Based on this normalized area index, the chip size has to be normalized by the technology process and the FFT size. This area normalized index can evaluate the area requirement per point FFT operation. In this work, we will use normalized area for the FFT design

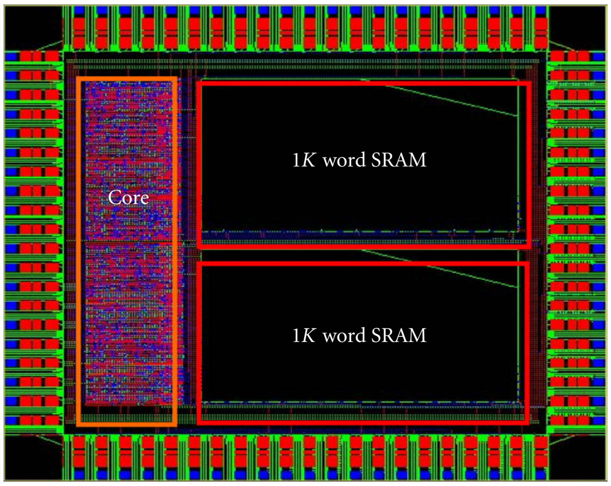


Figure 13: Layout of the FFT/IFFT processor.

based on 0.35 μm CMOS technology process to perform fair comparisons. That is,

$$\text{normalized area per FFT} = \frac{\text{Area}}{(\text{Technology}/0.35\,\mu\text{m})^2 \cdot \text{FFT Size}} \times 1000. \tag{6}$$

In addition, we also proposed the second normalized index about the power consumption, normalized power per FFT. Based on this normalized index, we can consider the effect of the supply voltage, FFT sizes, and operating frequency and then do comparisons with other FFT works very fairly, as follows:

$$\text{normalized power per FFT} = \frac{\text{Power}}{(\text{Voltage})^2 \cdot (\text{FFT Size}) \cdot \text{Frequency}} \times 1000. \tag{7}$$

According to these two normalize indexes, we compare the proposed scheme with several existing FFT designs in Table 5. As we can see, the area requirement and power consumption for this work is lower than the works of Baas [9] and Hui et al. [20]. On the other hand, the variable-length FFT/IFFT prototype processor can work at higher operating frequency than the works by Bidet et al. [17] and Jia et al. [21] and can meet various specifications in OFDM applications. It means that we can apply it on the multimode/multistandard OFDM communication systems under the very low implementation cost.

7. CONCLUSIONS

The OFDM technique is really very important in many wired and wireless communication systems. In order to implement the multimode/multistandard OFDM communication systems, we design and implement the variable-length FFT/IFFT processor. For the consideration of low-power consumption, we choose cached-memory architecture and

Table 4: FFT size, T_{FFT}, frequency, power consumption, and power efficiency.

Application	WLAN (IEEE 802.11a)	ADSL, VDSL, DAB	VDSL, DAB	VDSL, DAB, DVB-T
FFT Size (point)	64	512	1024	2048[a]
T_{FFT} (μs)	3.2	62	128	224
Operating frequency (MHz)	65	12.5	25	60
Power consumption (mW)	545	126	253	574

[a]Simulated results are based on $4K$-word embedded SRAM.

Table 5: Comparison of various FFT/IFFT processors.

Processor	CMOS tech. (μm)	FFT size (points)	Freq. (MHz)	Area (mm^2)	Normalized area per FFT	Normalized power per FFT	Multimode
This Work	0.35	64–2048	60	15.21	7.43	4.67	Yes
Baas [9]	0.6	1024	173	42.88	14.25	4.76	No
Bidet et al. [17]	0.5	8192	20	100	5.98	3.67	No
Colin [20]	0.6	64	36	62.4	331.77	434.03	No
Lihong [21]	0.6	8192	20	140	5.82	3.97	No

then we define the hardware architecture. We also adopted the newly proposed MSR-CORDIC algorithm to improve the rotation number and quantization noise for PE design without scaling operations. Moreover, the PE designs can have the low cost and low power of the advantage. The CLU and AG designs can control the number of points (64–2048) that FFT operations will perform. Finally, the variable-length FFT/IFFT processor is implemented with TSMC 0.35 μm CMOS technology process. The chip size is 15.21 mm^2 and power consumption ranges between 126–574 mW based on the operating points. The post-layout simulation results show that the variable-length prototype FFT/IFFT can handle 64–2048-point OFDM communication systems successfully.

ACKNOWLEDGMENT

This work was supported in part by the National Science Council, Taiwan, under Grants 91-2213-E-002-057 and 91-2215-E-002-044.

REFERENCES

[1] I. Kalet, "The multitone channel," *IEEE Trans. Communications*, vol. 37, no. 2, pp. 119–124, 1989.

[2] P. Chow, J. C. Tu, and J. M. Cioffi, "Performance evaluation of a multichannel transceiver system for ADSL and VHDSL services," *IEEE Journal on Communications*, vol. 9, no. 6, pp. 909–919, 1991.

[3] N. Al-Dhahir and J. M. Cioffi, "Optimum finite-length equalization for multicarrier transceivers," *IEEE Trans. Communications*, vol. 44, no. 1, pp. 56–64, 1996.

[4] R. V. Paiement, "Evaluation of single carrier and multicarrier modulation techniques for digital ATV terrestrial broadcasting," CRC Tech. Rep. CRC-RP-004, Communications Research Centre, Ottawa, Ontario, Canada, 1994.

[5] S. Hara, M. Mouri, M. Okada, and N. Morinaga, "Transmission performance analysis of multi-carrier modulation in frequency selective fast Rayleigh fading channel," *Wireless Personal Communications*, vol. 2, no. 4, pp. 335–356, 1996.

[6] ETSI EN 300 401 V1.3.3 (2001-05), "Radio Broadcasting Systems; Digital Audio Broadcasting (DAB) to mobile, portable and fixed receivers," 2001.

[7] R. Grisamore, G. N. Rangan, and E. E. Swartzlander Jr., "An efficient FFT processor for ADSL applications," in *Proc. 35th Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 1251–1255, Pacific Grove, Calif, USA, November 2001.

[8] S. H. Park, D. H. Kim, D. S. Han, K. S. Lee, S. J. Park, and J. R. Choi, "A 2048 complex point FFT processor for DAB systems," in *Proc. IEEE International Conference on VLSI and CAD*, pp. 309–312, Seoul, Korea, October 1999.

[9] B. M. Baas, "A low-power, high-performance, 1024-point FFT processor," *IEEE Journal of Solid-State Circuit*, vol. 34, no. 3, pp. 380–387, 1999.

[10] Z.-X. Lin and A.-Y. Wu, "Mixed-scaling-rotation CORDIC (MSR-CORDIC) algorithm and architecture for scaling-free high-performance rotational operations," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 2, pp. 653–656, Hong Kong, China, April 2003.

[11] W. Li and L. Wanhammar, "A pipeline FFT processor," in *Proc. IEEE Workshop on Signal Processing Systems*, pp. 654–662, Taiwan, China, October 1999.

[12] Y. H. Hu, "CORDIC-based VLSI architectures for digital signal processing," *IEEE Signal Processing Magazine*, vol. 9, no. 3, pp. 16–35, 1992.

[13] A.-Y. Wu and C.-S. Wu, "A unified view for vector rotational CORDIC algorithms and architectures based on angle quantization approach," *IEEE Trans. Circuits and Systems*, vol. 49, no. 10, pp. 1442–1456, 2002.

[14] R. Sarmiento, F. Tobajas, V. de Armas, R. Esper-Chaín, J. F. López, and J. A. Montiel-Nelson, "A CORDIC processor for FFT computation and its implementation using gallium arsenide technology," *IEEE Transaction on Very Large Scale Integrated Systems*, vol. 6, no. 1, pp. 18–30, 1998.

[15] C.-S. Wu and A.-Y. Wu, "A novel trellis-based searchingscheme for EEAS-based CORDIC algorithm," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 2, pp. 1229–1232, Salt Lake City, Utah, USA, May 2001.
[16] C.-S. Wu and A.-Y. Wu, "Modified vector rotational CORDIC (MVR-CORDIC) algorithm and architecture," *IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 6, pp. 548–561, 2001.
[17] E. Bidet, D. Castelain, C. Joanblanq, and P. Senn, "A fast single-chip implementation of 8192 complex point FFT," *IEEE Journal of Solid-State Circuit*, vol. 30, no. 3, pp. 300–305, 1995.
[18] H. Shousheng and M. Torkelson, "Designing pipeline FFT processor for OFDM (de)modulation," in *Proc. International Symposium on Signals, Systems, and Electronics (ISSSE '98)*, pp. 257–262, Pisa, Italy, September–October 1998.
[19] C. H. Chang, C. L. Wang, and Y. T. Chang, "A novel memory-based FFT processor for DMT/OFDM applications," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 4, pp. 1921–1924, Phoenix, Ariz, USA, March 1999.
[20] C. C. W. Hui, T. J. Ding, J. V. McCanny, and R. F. Woods, "A new FFT architecture and chip design for motion compensation based on phase correlation," in *Proc. International Conference on Application-Specific Systems, Architectures, and Processors*, pp. 83–92, Chicago, Ill, USA, August 1996.
[21] L. Jia, Y. Gao, J. Isoaho, and H. Tenhunen, "A new VLSI-oriented FFT algorithm and implementation," in *Proc. 11th Annual IEEE International ASIC Conference*, pp. 337–341, Rochester, NY, USA, September 1998.

Jen-Chih Kuo received the B.S. and M.S. degrees in electrical engineering from the National Taiwan University and Graduate Institute of Electronics Engineering, Taiwan, in 2000 and 2002, respectively. He is now with Genesys logic, Inc., Taipei, Taiwan. His research interests include VLSI architectures for DSP algorithms, adaptive signal processing, and digital communication systems.

Ching-Hua Wen received the B.S. degree in electrical engineering from the National Central University, Taiwan, in 2002. He is now at the Graduate Institute of Electronics Engineering of the National Taiwan University. His research interests include VLSI architectures for DSP algorithms, digital frequency synthesizers, and digital communication systems.

Chih-Hsiu Lin received the B.S. degree in mechanical engineering in 1998 from National Taiwan University, Taiwan, where he is currently working towards the Ph.D. degree in Graduate Institute of Electronics Engineering. His research interests include the VLSI implementation of communication systems and Multiuser detection for CDMA systems.

An-Yeu (Andy) Wu received the B.S. degree from National Taiwan University in 1987, and the M.S. and Ph.D. degrees from the University of Maryland, College Park, in 1992 and 1995, respectively, all in electrical engineering. During 1987–1989, he served as a Signal Officer in the Army, Taipei, Taiwan, for his mandatory military service. From August 1995 to July 1996, he was a member of the technical staff at AT&T Bell Laboratories, Murray Hill, NJ, working on high-speed transmission IC designs. From 1996 to July 2000, he was in the Electrical Engineering Department of National Central University, Taiwan. He is currently an Associate Professor in the Graduate Institute of Electronics Engineering and Department of Electrical Engineering, National Taiwan University, Taiwan. His research interests include low-power/high-performance VLSI architectures for DSP and communication applications, adaptive/multirate signal processing, and reconfigurable broadband access systems and architectures. Dr. Wu is currently serving as an Associate Editor for EURASIP Journal on Applied Signal Processing. He becomes the Associate Editor of the IEEE Transactions on Very Large-Scale Integration (VLSI) Systems in July 2003. He has served on the technical program committees of IEEE International Conferences such as ICIP, SiPS, AP-ASIC, SOC, and ISCAS.