國 立 交 通 大 學

電機與控制工程學系

博 士 論 文

高效能之管線式傅立葉轉換處理器之設計與實現

Design and Implementation of High-Effective Pipelined Processors for Discrete-Time Fourier Transform Applications

研 究 生：余 遠 渠

指導教授：林 進 燈

中華民國九十七年五月

高效能之管線式傅立葉轉換處理器之設計與實現

Design and Implementation of High-Effective Pipelined Processors for Discrete-Time Fourier Transform Applications

研 究 生：余 遠 渠　　Student: Yuan-Chu Yu

指導教授：林 進 燈　　Advisor: Chin-Teng Lin

國立交通大學

電機與控制工程學系

博士論文

A Dissertation

Submitted to Department of Electrical and Control Engineering

College of Electrical Engineering and Computer Science

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

In

Electrical and Control Engineering

May 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年五月

中文摘要

本篇論文針對傅立葉轉換，設計其高效能之管線式處理器。論文以四種不同之即時應用為範例來提出其對應之高效能設計，其包括：雙聲多頻偵測器在高通道密度之VoP 應用、多輸入多輸出之正交多頻的無線區域網路、多輸入點之長快速傅立葉轉換運算在手機之數位影像傳波系統應用、以及快速傅立葉正(反)轉換/二維數位餘弦轉換在下代手機之多媒體應用。針對這四種明顯不同之應用，本論文提出了六種特定之硬體導向設計，以達到最高效能之管線式處理器架構，其評估之指標包括: 單位時間輸出量、計算延遲時間、運算複雜度、硬體成本與硬體使用之利用率。在雙聲多頻偵測器之應用上，本論文採用：精簡式輸入序列架構、分散式記憶體以及柴比雪夫多項式為基準之改良式遞迴式轉換器，來達到低計算週期、高能量利用率之優點。所架構之單聲多頻偵測器單核心，可在相同之運算速度及運算時間內，達到雙倍之資料運算量。對於 2×2 以及 4×4 多輸入多輸出之正交多頻的無線區域網路，本論文提出兩種高效能之快速傅立葉正(反)轉換處理器：積數 2/8 之多回授路徑架構(R28MDF)與積數 2/8 之多延遲整流路徑架構(R28MDC)。依據精簡式之基數 8 快速傅立葉轉換單元(R8-FFT)，配合先寫後讀(MAW)之技巧，此兩架構達到了 100%之蝴蝶器利用率，同時更在單位時間內達到高輸出量已滿足 2×2 以及 4×4 多輸入多輸出之正交多頻之無線區域網路需求。針對多輸入點之長快速傅立葉轉換運算應用上，本論文提出兩個新式架構：基數 4^2 單一迴授路徑架構與基數 4^3 單一迴授路徑架構，其以較少之基數 4 理論來達到高基數 16 與基數 64 之低運算複雜度效能。在跟其他數個已存在之管線式處理器比較後，可證明本論文所提出之架構，以最少之硬體成本達到最高之硬體使用率，因此達到了高效能之應用需求。最後根據基數 4^2 單一迴授路徑架構,配合區段移位暫存器與翻轉移位暫存器架構,架構了一”三模處理器”來支援 256 點之快速傅立葉正(反)轉換運算與二維數位餘弦轉換運算。同樣地，在跟其他數個現存之管線式處理器比較後，可證明本論文所提出之架構，以最少之硬體成本達到最高之硬體使用率，因此達到了高效能之應用需求。在本論中六個處理器皆以用 TSMC 0.13μm CMOS 製程完成實現與驗證，根據實現結果與嚴謹之比較，我們可證明本文所提出之 RDFT、R28MDF/R28MDC、R4^2SDF/ R4^3SDF 與三模處理器，在雙聲多頻偵測器、多輸入多輸出之正交多頻的無線區域網路、多輸入點之長快速傅立葉轉換運算、下代手機之多媒體應用上皆達到高處理效能之優點。

Design and Implementation of High-Effective Pipelined Processors for Discrete-Time Fourier Transform Applications

Student：Yuan-Chu Yu Advisor：Chin-Teng Lin

Department of Electrical and Control Engineering
National Chiao-Tung University

ABSTRACT

In this thesis, the design and implementation of effective pipeline processors for Fourier transform are presented. Four different real-time applications are introduced, which includes dual tone multi-frequency (DTMF) detector in the high channel density voice over packet (VoP) application, multiple-input multiple-output orthogonal frequency division multiplexing (MIMO-OFDM) wireless LAN (WLAN) system, long-length based FFT/IFFT computations in digital video broadcasting－handheld (DVB-T) standard and FFT/IFFT/2D-DCT computations in next generation mobile multimedia applications. According to these four standards, six specific hardware-orientated designs for most effective pipeline processors have been proposed in terms of throughput, computation latency, computation complexity, hardware cost and hardware utilization.

For the DTMF standards, one low-computation cycle and power-efficient recursive DFT/IDFT processor adopting a hybrid of input strength reduction, the Chebyshev polynomial, and register-splitting schemes has been proposed. Appling this novel low-computation cycle architecture, we could double the throughput rate and the channel density without increasing the operating frequency for the DTMF detector in the high channel density VoP application. Two effective FFT/IFFT processors, namely adix-2/8 multiple-path delay feedback (R28MDF) based and raidx-2/8 multiple-path delay commutator (R28MDC) based FFT/IFFT processors for the 2×2 and 4×4 MIMO-OFDM WLAN systems, respectively. By applying the retrenched 8-point FFT (R8-FFT) unit combined with the proposed multiplication-after-write (MAW) method, the R28MDF and R28MDC architectures resulted in 100% butterfly utilization and an appropriate throughput rate with few hardware resources for the 2×2 and 4×4 MIMO-OFDM applications, respectively. For the long-length based FFT/IFFT computations, two novel radix-4^2 single-path delay feedback (R4^2SDF) design and radix-4^3 single-path delay feedback (R4^3SDF) design with the low computational complexities of the radix-16 and radix-64 algorithms and the low hardware requirement of the radix-4 algorithm achieve the smallest hardware cost and the highest hardware utilization among the tested architectures and thus has the highest efficiency. Base on the effective R4^2SDF architecture with the segment shift register (SSR) and overturn shift register (OSR) structure, the proposed triple-mode processor not only supports both 256-point FFT/IFFT and 8×8 2-D DCT computations, but also has the smallest hardware requirement and largest hardware utilization among the tested architectures for the FFT/IFFT computation, and thus has the highest cost efficiency.

In this thesis, six processors all implemented under TSMC 0.13μm CMOS process. According to the comprehensive comparisons and implementation results, we could demonstrate that the proposed RDFT, R28MDF/R28MDC, R4^2SDF/ R4^3SDF and Triple-Mode designs achieve the high effective advantages for DTMF, MIMO-OFDM WLAN, DVB-T and next-generation applications.

致謝

本論文的整個研究過程，需要感謝的人實在太多了，無論是碩士班的學弟還是博士班的同學與學長們，對於我都給予我相當多的支持與鼓勵，讓我在博士班的時期能不斷地精進。

最感謝當然是指導教授 林進燈博士的悉心指導，在研究的方向總是給我最正確的方向與建議，儘管指導教授再忙碌，也從不望給予我鼓勵與指導，且老師也給我相當大的彈性，讓我能學習如何面對及解決問題的正確態度與方法。另外要感謝 邱創乾教授千里迢迢來擔任我的口試委員召集人，教授們的建議與指導，更讓本論文的內容更加充實與完整。

其次是要感謝交通大學資訊工程系的 范倫達教授，在我的研究內容與方向上給我相當多的建議與幫助，並在我博士班時期遇到的所有困難與低潮時，都給予最大的協助與體諒。在這段時間和我共同度過許許多多難忘的回憶。同時也感謝 義隆電子股份有限公司的 顏國隆副總的賞識與補助，讓我可在上班的閒暇之餘，完成如此艱鉅的博士班求學過程。

最後最要感謝的是默默支持我的老婆，以及母親、弟弟與女兒給予我精神及物質上的一切支援，也感謝其他親朋好友的關心與鼓勵。你們的關心與支持，才是使我保持研究的動力與精神來源。

謹以本論文獻給我的家人及所有關心我的師長與朋友們。

Contents

List of Figures

List of Tables

Chapter 1 Introduction

The increased demand for communication, multimedia, and other consumer products has created the need for low-cost, low-power consumption and high throughput based processor that can use Fourier transforms for their signal processing or data manipulation. The discrete Fourier transform (DFT) is an equation for converting time domain data into frequency domain data [1]. Discrete means that the signal is sampled in time rather than being continuous. Therefore, DFT is an approximation for the continuous Fourier transform [2]. The DFT equation, unlike the continuous Fourier transform, covers a finite time and frequency span. Base on the requirements of the DFT results, there are possible two categories for the effective algorithms of DFT computations: 1) fast Fourier transform (FFT) algorithm, 2) recursive algorithm. FFT based algorithms are a group of algorithms for significantly speeding up the computation of the DFT, when all N points of DFT results are required. The most widely known of these algorithms is attributed to Cooley and Tukey [3] and is used for a number of point N equal to a power-of-two. In the realistic world, many applications require spectrum analysis only over a subset of the N center frequencies via the DFT computation instead of the overall results of the FFT. An effective derivative of DFT is the recursive based algorithm, which emerges better performance than the FFT algorithm when only some sparse DFT results need to be obtained by completing a single complex DFT spectral bin value for every N input time instances. The most famous of the recursive algorithms is the Goertzel algorithm [4], which use the periodicity properties to reduce DFT computations. Base on the required portions of DFT results, two effective DFT processors could be found: 1) FFT based processor, 2) recursive based processor. In this study, one high effective recursive processor has been presented. Base on the different requirements, five different pipeline FFT/IFFT processors are also presented in this work.

1.1 Motivation

Many researchers have concentrated on designing an optimized reconfigurable DSP processor to achieve a high processing rate and low power consumption in next-generation mobile multimedia applications [5][6]. The software based architecture such as the co-processor and dual-MAC designs have been proposed by Chai *et al.* [5] and Kolagotla *et al.* [6], respectively. However, they induce the large chip size because of the high flexibility. Vorbach *et al.* have also presented hardware-based concepts such as the processing element (PE) array [7], which achieves a high processing rate with reasonable flexibility. However, the processing kernel has the flaw of a low utilization rate with a large array memory and muti-MACs, leading to poor cost efficiency. The specific ASIC based design on a fast computation algorithm provides high cost efficiency [8]-[10]. Base on the different real-time applications, some design decisions for ASIC based FFT processor should be made following with the different specification:

- *Required portions of DFT results*: The primary advantage of recursive based algorithms is that it allows a subset of the DFT's N output terms to be efficiently calculated. Considering the computation complexity, the direct evaluation of DFT of all N values requires a total of N^2 complex multiplications and $N(N\text{-}1)$ complex additions. If only M values of N DFT results are required, the computation complexity of Goertzel and radix-2 based FFT algorithm are NM and $N\log_2 N$, respectively. It is obviously that the computation saving of radix-*2* based FFT algorithm is not significant —less than a factor of two. Then, the Goertzel algorithm demonstrates the good efficient for certain applications, such as: the dual tone multi-frequency (DTMF) standards [11-16] for voice over packet (VoP) network [17-19], discrete multi-tone equalizer of multi-carrier modulation system [20, 21], and speed detection.
- *Number of FFT channels:* Future broadband wireless access systems including wireless LANs (WLAN) and fourth-generation (4G) mobile radio systems need much higher spectral efficiency and service quality than the current standards do [22, 23]. A multiple-input-multiple-output (MIMO) wireless system has been extensively studied recently due to the potential for raising system capacity [24, 25, 26]. The orthogonal frequency division multiplexing (OFDM) modulation scheme not only decreases the receiver complexity, but also improves the performance on highly dispersive channels. An especially promising candidate for the next-generation fixed and mobile wireless systems is the combination of MIMO technology with OFDM, called the MIMO-OFDM

system. A MIIMO-OFDM system with *k* antennas in the transmitter and the receiver comprises *k* OFDM baseband processors working in parallel, and thus requires *k* FFT processors, one for each antenna [24-26]. Then, a high throughput FFT processor, which could compute the multi-channel FFT computations, would be required.

- *Transform length of FFT computation*: The size of the transform will directly affect frequency resolution, memory requirements, and the speed at which the computation can be done. In the realistic world, many applications require the FFT/IFFT implementations that can perform long-length computations while exhibiting low cost, low power consumption and high throughput. The long-length based FFT/IFFT processor has been widely applied in many real time applications, such as: DVB-H(Digital Video Broadcasting－Handheld)[27, 28], VDSL(Very-high-speed Digital Subscriber Line) [29], and audio measurement [30]. Since such long-length FFT computations are rather time-consuming, the efficient FFT processors are necessary to meet the real time operations. Furthermore, the handheld devices include multimedia mobile phones with color displays as well as personal digital assistant (PDA) and pocket PC, which should consider some specific advantages — small, lightweight, portable, battery-powered devices.
- *Number of dimension*: All multidimensional FFTs are done as a sequence of one-dimensional FFTs. The importance of knowing how many dimensions (one, two, or three, usually) there are determines how many FFTs will be need and how the data must be organized to do the multiple dimensions. This will affect chip processing load and the choice of architecture. To improve the radix-2 based FFT algorithm, He *et al.* [31] has presented radix-2^2 and radix-2^3 algorithms for the higher computation efficiency. Then, the design in [31] achieves the high hardware utilization and low hardware resource usage.
- *Algorithm construction*: The algorithm used will affect the computational complexity the algorithm requires and computation speed the design does. The low radix based algorithm is well known to have higher multiplicative complexity than the high radix based algorithm. Notably, the design with the highest complex multiplicative complexity has the highest power consumption [26, 28, 31-33].
- *Architectures*: Many researches were concentrated on the efficient FFT realizations [26, 31, 34-36]. The appropriated algorithm and architecture for the FFT processor should be chosen trading off its processing speed and its chip cost. The pipeline architecture processes regularity, modularity, local connection, and high throughput rate with lower

clock frequency [37]. Furthermore, pipeline FFT processor is characterized by non-stopping processing on a clock frequency of the input data sampling. An analysis has depicted that a unique operating frequency, which is close or equivalent to the sampling frequency is preferable to the FFT processor when the power consumption is confined by the application environment, such as handheld communications [26, 31, 32, 34, 38]. Basically, there are mainly two different pipeline architectures: multipath delay commutator (MDC) architectures [33, 36, 39, 40] and single-path delay feedback (SDF) architectures [31, 32, 34, 35, 42, 43]. The SDF architectures are well known to be more efficient than MDC architectures in terms of memory utilization since the butterfly output share the same storage with its input [31, 32, 34]. Therefore, this investigation focuses on the "hardware-oriented" pipeline architecture, in which the arithmetic operations can be tightly scheduled for effective hardware utilization.

1.2 Objectives

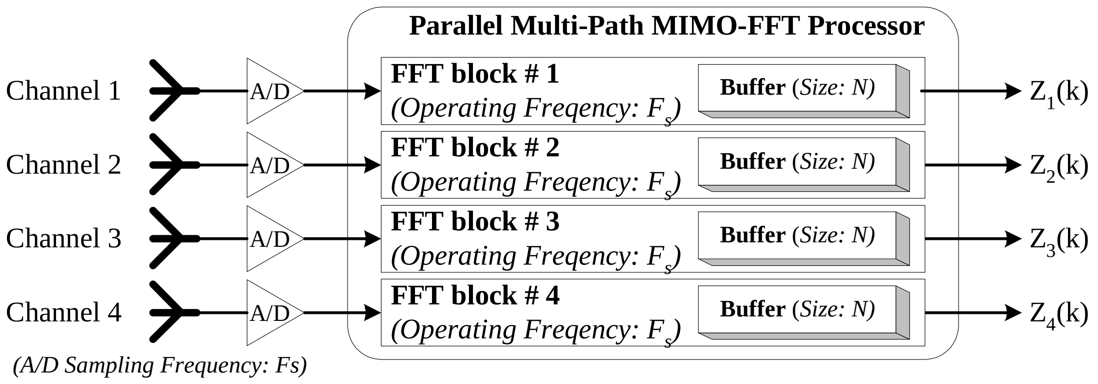
The objectives of this thesis are to propose the high effective pipeline processors for the DFT computations in different real-time applications. Four different applications have been taken into consideration, which are recursive based DFT computation in DTMF standard [12-15], multiple-input multiple-output orthogonal frequency division multiplexing (MIMO-OFDM) wireless LAN (WLAN) [22, 23], long-length based FFT/IFFT computations in digital video broadcasting－handheld (DVB-T) standard [27, 28] and FFT/IFFT/2D-DCT computations in next generation mobile multimedia applications [5-7, 44]. The objective descriptions of these four designs are provided as below:

1. *Recursive DFT/IDFT Design*: The Goertzel algorithm has been widely applied to the dual tone multi-frequency (DTMF) standards [11]-[16] for voice over packet (VoP) network [17]-[19] to compute the interested spectra, the discrete multitone equalizer of multicarrier modulation system [20]-[31], and speed detection. Considering the state-of-the-art applications, the high channel-density dual-tone detector [17]-[19] is demanded. Some advanced DTMF detectors for the high density VoP network application have been realized by one embedded DSP processor [12]-[14], [17]-[19]. Although, the DSP processor based design could keep the maximum flexibility, it may

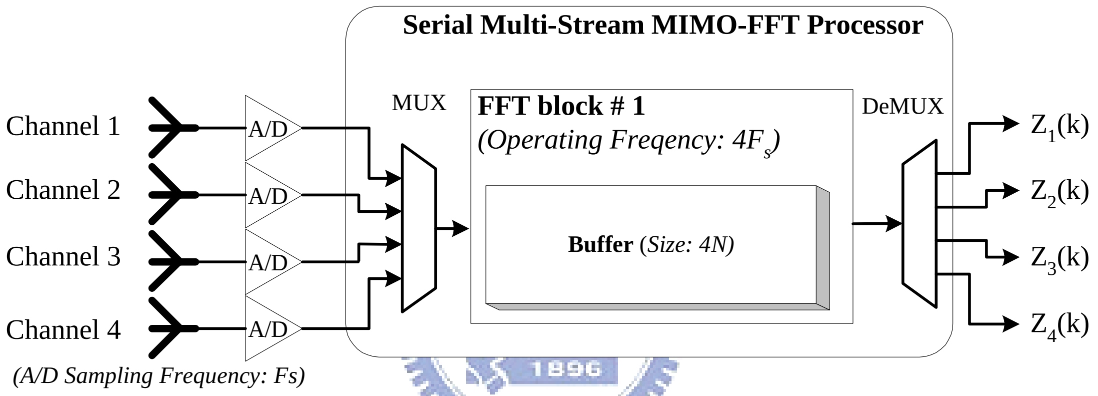
not meet the cost effective considerations. On the other hand, the DSP processor based design may lose the advantages of high-throughput, low power, and small area compared with the application-specific integrated circuits (ASIC) designs [45]. In [13], the DSP processor based DTMF detectors needs a large amount of memory to decode only 24 channels, which requires 800 words data memory and 1000 words program memory with 16-bit wordlength for each words. Also, it has to operate on the higher frequency of 24 MHz. For the purpose of optimizing the whole system performance and cost, much research [46]-[53] has concentrated on the dedicated core design. In [15]-[17], the recursive expressions for the DCT computation that are suitable for VLSI implementation are presented. It is worth noticing that the recursive algorithms are solely used to design recursive DCT architectures rather than the recursive DFT architectures in [46]-[48]. In the past two decades, several recursive DFT algorithms and architectures have been explored [49]-[53]. Compared with the conventional second-order recursive DFT/IDFT architecture, Van *et al*. [51] utilized resource-sharing and register-splitting schemes to reduce two multipliers and speedup the computation, respectively. Yang *et al.* [52] proposed two unified IIR filter structures to save the hardware cost for the DFT computation. Nevertheless, neither Van *et al.* [51] nor Yang *et al.* [52] improve the computation cycle. In [53], Fan *et al.* applied the previous proposed method to reduce the computation cycles but the performance is limited. On the other hand, Fan *et al.* only proposed the recursive DFT algorithm but the IDFT algorithm is not yet ready in [53]. In essence, a short description of the proposed algorithm has been presented in the associated conference [54, 55]. In this thesis, the detailed descriptions of a high-performance and power-efficient VLSI algorithm and architecture by the hybrid of input strength reduction scheme, Chebyshev polynomial, and register-splitting scheme for the DTMF application have been fully provided. The derived recursive algorithm and devised architecture [54, 55] possesses the following features: low-computation cycle (i.e., high throughput) and power efficiency at the expense of slightly increased area overhead compared with the existing recursive DFT/IDFT structures.

2. *MIMO-OFDM FFT design*: Future broadband wireless access systems including wireless LANs (WLAN) and fourth-generation (4G) mobile radio systems need much higher spectral efficiency and service quality than the current standards do [22, 23]. A multiple-input-multiple-output (MIMO) wireless system has been extensively studied recently due to the potential for raising system capacity [24-26]. The orthogonal

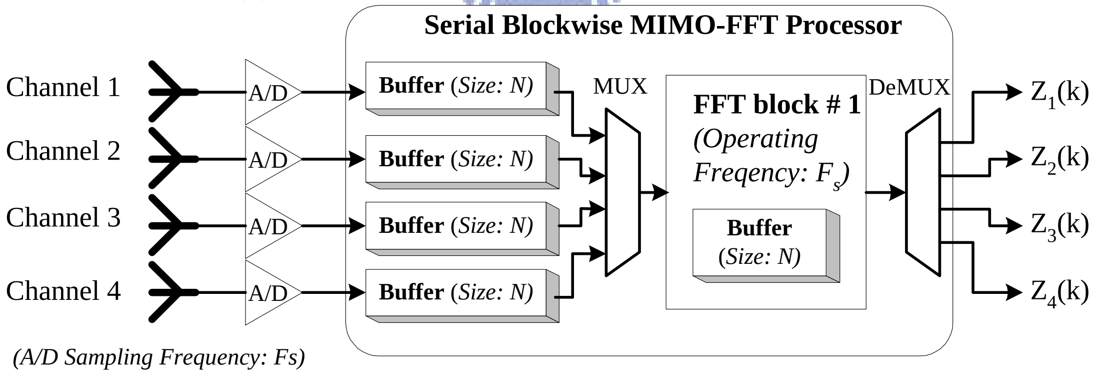
frequency division multiplexing (OFDM) modulation scheme not only decreases the receiver complexity, but also improves the performance on highly dispersive channels. An especially promising candidate for the next-generation fixed and mobile wireless systems is the combination of MIMO technology with OFDM, called the MIMO-OFDM system. A MIIMO-OFDM system with *k* antennas in the transmitter and the receiver comprises *k* OFDM baseband processors working in parallel, and thus requires *k* FFT processors, one for each antenna [24-26]. Because of the high throughput requirements of the FFT computation in the MIMO-OFDM system, three 4×4 MIMO-FFT architectures, parallel multi-path architecture, serial multi-stream architecture and serial blockwise architecture, as depicted in Fig. 1(a)-(c), respectively, have been presented [25]. A parallel multi-path architecture includes *k* FFT blocks for *k* antennas, as depicted in Fig. 1(a). The figure indicates that the area cost of parallel multi-path based system rises linearly with the number of antennas (i.e. *k* times the FFT block area). Conversely, the serial multi-stream architecture and serial blockwise architecture only requires one FFT block to handle the concurrent computation of *k* antennas. However, the serial multi-stream architecture applies one lower throughput rate FFT processor embedded with the *k* times buffer size for intermediate computation, as depicted in Fig. 1(b). For *k* channel computation, the serial multi-stream architecture must operate at a higher clock frequency than sampling data frequency of F_s to satisfy the higher throughput requirements. Analytical results indicate that the operating frequency of serial multi-stream based system grows linearly with the number of antennae (i.e. *k* times the sampling frequency of F_s). Based on the serial blockwise FFT architecture, the input data of the FFT block can be provided in parallel with *k* embedding input buffer, as depicted in Fig. 1(c). Applying one higher throughput rate FFT processor, the serial blockwise FFT based processor can complete *k* channel FFT computations concurrently. Among these three architectures, the serial blockwise architecture only requires one FFT block operating at the same clock frequency with the data sampling frequency of F_s. An analysis has depicted that a unique operating frequency, which is close or equivalent to the sampling frequency of F_s, is preferable to the FFT processor when the power consumption is confined by the application environment, such as mobile communications [26, 31, 38, 56, 57]. Considering the memory cost, the serial blockwise architecture should slightly increase the cost with one extra buffer of size *N* than other architectures. However, the memory cost problem for serial blockwise architecture becomes increasingly minor when the number of antennae in the MIMO-OFDM system is larger. Consequently, the serial blockwise-based MIMO-FFT architecture applies single FFT block to achieve the appropriate throughput and minimizes power consumption for MIMO-OFDM WLAN applications.


Parallel Multi-Path MIMO-FFT Processor
Channel 1
Channel 2
Channel 3
Channel 4
A/D
FFT block # 1 (Operating Freqency: F_s)
FFT block # 2 (Operating Freqency: F_s)
FFT block # 3 (Operating Freqency: F_s)
FFT block # 4 (Operating Freqency: F_s)
Buffer (Size: N)
$Z_1(k)$
$Z_2(k)$
$Z_3(k)$
$Z_4(k)$
(A/D Sampling Frequency: Fs)

(a) Parallel multi-path MIMO-FFT architecture.


Serial Multi-Stream MIMO-FFT Processor
Channel 1
Channel 2
Channel 3
Channel 4
A/D
MUX
FFT block # 1 (Operating Freqency: $4F_s$)
Buffer (Size: 4N)
DeMUX
$Z_1(k)$
$Z_2(k)$
$Z_3(k)$
$Z_4(k)$
(A/D Sampling Frequency: Fs)

(b) Serial multi-stream MIMO-FFT architecture.



(c) Serial blockwise MIMO-FFT architecture.

Fig. 1: MIMO-FFT architectures.

3. *Long-Length FFT Design*: The FFT and IFFT are essential in the field of digital signal processing (DSP) and communication systems. In the realistic world, many applications require the FFT/IFFT implementations that can perform long-length computations while exhibiting low cost, low power consumption and high throughput. The long-length

based FFT/IFFT processor has been widely applied in many real time applications, such as: DVB-H(Digital Video Broadcasting－Handheld)[27, 28], VDSL(Very-high-speed Digital Subscriber Line) [29], and audio measurement [30]. DVB-H is a digital broadcast standard offering high data rate audio/video content delivery to handheld devices, which requires a 4096-point FFT computation (i.e. 4k mode) for the flexible networking design in single frequency networks (SFNs) [27, 28]. The VDSL transceiver and audio analyzer need to involve the complicated FFT computations, where the transform length is also 4096-point [29, 30]. Since such long-length FFT computations are rather time-consuming, the efficient FFT processors are necessary to meet the real time operations. Furthermore, the handheld devices include multimedia mobile phones with color displays as well as personal digital assistant (PDA) and pocket PC, which should consider some specific advantages — small, lightweight, portable, battery-powered devices.

4. *Triple-mode reconfigurable FFT/IFFT/2-D DCT design*: generation mobile multimedia applications, including mobile phones and personal digital assistant (PDAs), require much sufficiently high processing power for multimedia applications. Multimedia applications include video/audio codecs, speech recognition and echo cancellers. The speech recognition requires the speech extraction and autocorrelation coefficient computations [58] in the voice command application. The video codec is the most challenging element of a multimedia application, since it requires much processing power and bandwidth. Hence, a flexible and low cost pipeline processor with the superiority of high processing rate is required to realize necessary computation-intensive algorithms, such as 256-point FFT/IFFT and 8×8 2-D DCT [5]-[7]. Additionally, a major integration challenge is to design the digital baseband and accompanying control logic. The WiMAX baseband is constructed around orthogonal frequency division multiplexing (OFDM) technology requiring high processing throughput. The fixed, IEEE 802.16e [44], version of WiMAX also needs a 256-point FFT computation. Many researchers have recently concentrated on designing an optimized reconfigurable DSP processor to achieve a high processing rate and low power consumption in next-generation mobile multimedia applications [5][6]. The software based architecture such as the co-processor and dual-MAC designs have been proposed by Chai *et al.* [5] and Kolagotla *et al.* [6], respectively. However, they induce the large chip size because of the high flexibility. Vorbach *et al.* have also presented hardware-based concepts such as the processing element (PE) array [7], which achieves a high processing rate with

reasonable flexibility. However, the processing kernel has the flaw of a low utilization rate with a large array memory and muti-MACs, leading to poor cost efficiency. The specific ASIC based design on a fast computation algorithm provides high cost efficiency [8]-[10]. Tell *et al.* [8] presented the FFT/WALSH/1-D DCT processor for multiple radio standards of the upcoming 4th generation wireless systems. Conversely, some designs [8]-[10] only support 1-D DCT computation, and have no 2-D DCT support. However, 2-D DCT is desirable for the video compression among wireless communication applications. This study not only presents a single reconfigurable architecture for the 256-point FFT/IFFT modes and the 8×8 2-D DCT mode, but also achieves high cost-efficiency in portable multimedia applications.

1.3 Contributions

For the purpose of supporting these four applications, six ASIC based pipeline processors, namely recursive DFT/IDFT (RDFT) based processor, radix-2/8 multiple-path delay feedback (R28MDF) based processor, radix-2/8 multiple-path delay commutator (R28MDC) based processor, radix-4^2 single-path delay feedback (R4^2SDF) based processor, radix-4^3 single-path delay feedback (R4^3SDF) based processor and reconfigurable triple-mode FFT/IFFT/2-D DCT processor, have been presented in this thesis. The contributive descriptions are presented as below:

1. *RDFT Design*: Based on the proposed RDFT architecture, one high-throughput (i.e. high channel density) and power-efficient DTMF detector has been proposed. For the purpose of achieving the high power efficiency, we perform the bit level SNR simulation to decide the best configuration for the DTMF detector system. The results show that the proposed design only needs 9-bit word-length, which is one-bit less than the second order Goertzel structure, to land the satisfactory resolution under 15 dB SNR environment. In this paper, the resulting DTMF detector uses 12-bit word-length, where the additional 3 bits are used for design margins so as to obtain better performance. On the other hand, the novel design saves 4-bit cost compared with the 16-bit based DSP processor design [12]-[14]. In summary, the proposed DTMF structure not only saves more area cost, but also reduces the power consumption due to the register-splitting

scheme [51] and a smaller word-length requirement. Most importantly, the computation cycles can be reduced to 50% and thus a double throughput rate and channel density can be easily obtained without increasing the operation frequency. Our proposed DFT/IDFT chip is able to offer over 128-channel telephone signals for the high channel density DTMF detector [16] without any DSP processor inside. Each channel consumes 9.77 uW under 1.2V@20 MHz in TSMC 0.13 1P8M CMOS process. This is a significant contribution, as the high channel density and low power characteristics are demanded for the communication systems.

2. *R28MDF and R28MDC Design*: This investigation presents two new efficient designs, R28MDF based and R28MDC based FFT/IFFT processors for the 2×2 and 4×4 multiple-input multiple-output orthogonal frequency division multiplexing (MIMO-OFDM) wireless LAN (WLAN) system, respectively. The novel radix-2/8 algorithm reduces the half constant multiplier requirement in the proposed retrenched 8-point FFT (R8-FFT) unit compared with that of the conventional radix-2/8 algorithm, and has low multiplicative complexity as a radix-8 based algorithm. By applying the R8-FFT unit combined with the proposed multiplication-after-write (MAW) method, the R28MDF and R28MDC architectures resulted in 100% butterfly utilization and an appropriate throughput rate with few hardware resources for the 2×2 and 4×4 MIMO-OFDM applications, respectively. Implementation results indicate that two chips consume only 19.42mW and 23.57mW under 1.2V@20 MHz in a TSMC 0.13μm 1P8M CMOS process. The comparison results among the existing 64-point FFT/IFFT processor architectures are comprehensively discussed. The architecture analyses and chip implementation indicate that the proposed FFT/IFFT processor architectures are suitable for MIMO-OFDM WLAN systems.

3. *R4^{2}SDF and R4^{3}SDF Design*: In this investigation, we proposes the novel radix-4^2 and radix-4^3 algorithms with the low computational complexities of the radix-16 and radix-64 algorithms and the low hardware requirement of the radix-4 algorithm. Base on the multiplierless radix-4 butterfly structure, the proposed R4^{2}SDF design and R4^{3}SDF design support the 4096-point FFT/IFFT computations. Moreover, the retrenched constant multiplier and eight-folded complex multiplier structures are adopted to decrease the multiplier cost and the coefficient ROM size with the complex conjugate symmetry rule and subexpression elimination technology. To further decrease the chip cost, a finite word-length analysis is provided to indicate that the proposed R4^{2}SDF and R4^{3}SDF architectures only require 14 and 13-bit internal word-length to achieve 40dB

SNR performance in the 4096-point FFT/IFFT computation. The comprehensive comparison results indicate that the proposed R4^3SDF design has the smallest hardware cost and the highest hardware utilization among the tested architectures for the FFT/IFFT computation, and thus has the highest efficiency. The implementation results show that the proposed R4^2SDF and R4^3SDF based 4096-point pipeline FFT/IFFT processors only consumes 6.3725 and 5.985 mW@20 MHz at 1.2V supply voltage in TSMC 0.13 μm CMOS process.

4. *The triple-mode reconfigurable FFT/IFFT/2D-DCT Design*: Applying the R4^2SDF architecture with the specific linear mapping of common factor algorithm (CFA), the proposed triple-mode design supports both 256-point FFT/IFFT and 8×8 2-D DCT modes following with the high efficient feedback shift registers architecture. The segment shift register (SSR) and overturn shift register (OSR) structure are adopted to minimize the register cost for the input re-ordering and post computation operations in the 8×8 2-D DCT mode, respectively. Moreover, the retrenched constant multiplier and eight-folded complex multiplier structures are adopted to decrease the multiplier cost and the coefficient ROM size with the complex conjugate symmetry rule and subexpression elimination technology. To further decrease the chip cost, a finite wordlength analysis is provided to indicate that the proposed architecture only requires a 13-bit internal wordlength to achieve 40dB SNR performance in 256-point FFT/IFFT modes and high digital video (DV) compression quality in 8×8 2-D DCT mode. The comprehensive comparison results indicate that the proposed cost effective reconfigurable design has the smallest hardware requirement and largest hardware utilization among the tested architectures for the FFT/IFFT computation, and thus has the highest cost efficiency. The derivation and chip implementation results show that the proposed pipeline 256-point FFT/IFFT/2-D DCT triple-mode chip consumes 22.37mW@100 MHz at 1.2V supply voltage in TSMC 0.13μm CMOS process, which is very appropriate for the RSoCs IP of next-generation handheld devices.

1.4 Organization

The remainder of this thesis is organized as follows.

Chapter 2 reviews the literature of the work presented in this thesis and four topics are reviewed. The first topic is a review of the Goertzel algorithm and respective hardware architecture. The second topic is a review of mixed-radix based FFT algorithms. The third topic is a comparative review of high-radix based FFT algorithms. The final topic is a review the DCT algorithm.
Chapter 3 describes a new recursive DFT/IDFT algorithm and architecture by the hybrid of input strength reduction, Chebyshev polynomial, and register-splitting schemes is revealed. Applying this new architecture, the DTMF application has been demonstrated. After the bit-level SNR simulation, the 212/106-point DFT/IDFT chip has been successfully implemented for the DTMF detector system. Furthermore, the comparison results are tabulated in terms of the amount of computation cycles for each output as well as *N*-point DFT/IDFT, the maximum number of the channel density, the clock period, and the number of real multipliers.

Chapter 4 describes a modified radix -2/8 FFT/IFFT algorithm. Using this mixed-radix based algorithm, we discuss the corresponding R28MDF and R28MDC fabrics and the detailed timing considerations. Furthermore, the implementation issues are discussed. Finally, the comparison results of the 64-point FFT/IFFT architectures for the 2×2 and 4×4 MIMO-OFDM system have been summarized.

Chapter 5 describes a new radix-4^2 and radix-4^3 FFT/IFFT algorithms. Applying these algorithms, the proposed R4^2SDF and R4^3SDF VLSI architectures could be demonstrated. Base on the finite word-length analysis, we could prove that the proposed architectures achieve the satisfactory system performance. Furthermore, the comparison results in terms of hardware utilization and cost demonstrate the high cost-efficiency of the proposed architectures. The chip implementation is also presented.

Chapter 6 describes a new triple-mode radix-4^2 FFT/IFFT and 8×8 2D DCT algorithm. Using the proposed radix-4^2 algorithm, the proposed R4^2SDF based FFT/IFFT/2-D DCT pipeline architecture is demonstrates. The finite wordlength analysis indicates that the proposed architecture achieves the required system performance in both 256-point FFT/IFFT and 8×8 2-D DCT modes with the lowest hardwire cost. According to the comparison results in terms of hardware utilization and cost, we could demonstrate the high cost-efficiency of the proposed architecture. Finally, the chip implementation is presented.

Chapter 2 Literature Review

The research work described in this thesis pertains to the design and realization of high effective pipeline processor for DFT/IDFT computations in different applications as discussed in Chapter 1. In this chapter, we consider a number of algorithms for computing the DFT. The algorithms vary in efficiency, but all of them require fewer multiplications and additions than does direct evaluation of DFT. This chapter will review four different topics relating to four different applications as discussed in the chapter 1. First, a review of the Goertzel algorithm and respective hardware architecture is presented. Second, a r eview of mixed-radix based FFT algorithms is presented. Third, a comparative review of high-radix based FFT algorithms is discussed. Finally, the algorithm mapping between FFT and DCT is detail reviewed.

2.1 The Goertzel Algorithm

In this section, we first discuss the Goertzel's algorithm [4], which requires computation proportional to N^2, but with a smaller constant proportionality than that of the direct computation of DFT. Notably, the Goertzel's algorithm is that it is not restricted to computation of the DFT, but is in fact equally valid for the computation of any desired set of samples of the Fourier transform of a sequence. Adopting the periodicity of the sequence W_N^{kn} , the Goertzel algorithm efficiently reduce the computation complexity of DFT computation.

2.1.1 The Recursive DFT Algorithm

Given input sequence and DFT output sequence denoted as $x[n]$ and $X[K]$, respectively, the N-point DFT can be defined as

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot W_N^{kn}, \tag{1}$$

where $W_N = e^{-j2\pi/N}$. The Goertzel algorithm [4] making use of the periodicity of the sequence W_N^{kn} can be used to reduce computation. For convenience of deriving a new architecture, we begin a review of the recursive DFT expression based on Goertzel algorithm by noting that

$$W_N^{-kN} = e^{-j(2\pi/N)Nk} = e^{j2\pi k} = 1. \tag{2}$$

Because of Eq. (2), we may multiply the right side of Eq. (1) by W_N^{-kN} without affecting the equation. Thus,

$$X[k] = W_N^{-kN} \sum_{r=0}^{N-1} x[r] \cdot W_N^{kr} = \sum_{r=0}^{N-1} x[r] \cdot W_N^{-k(N-r)}. \tag{3}$$

In order to simplify the final expression, let us define the sequence

$$y_k(n) = \sum_{r=-\infty}^{\infty} x[r] \cdot W_N^{-k(n-r)} \cdot u[n-r]. \tag{4}$$

From Eqs. (3) and (4) and the fact that $x[n]=0$ for $n<0$ and $n \geqq N$, it follows that

$$X[k] = y_k[n]\big|_{n=N}. \tag{5}$$

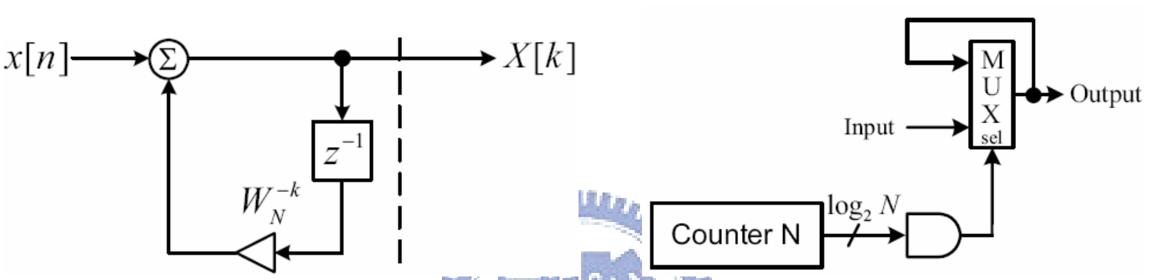
Eq. (4) can be interpreted as a discrete convolution of the finite-duration sequence $x[n]$, $0 \leqq n \leqq N-1$, with the $W_N^{-kn} u[n]$. As a consequence, $y_k(n)$ can be regarded as the response of a system with impulse response $W_N^{-kn} u[n]$ to a finite-length input $x[n]$. In particular, $X[k]$ is the value of the output when $n=N$. Taking the z-transform of Eq. (4), we can obtain the first-order transfer function as

$$H_k[z] = \frac{1}{1 - W_N^{-k} z^{-1}}. \tag{6}$$

It is possible to retain this simplification while reducing the number of multiplications by a factor of 2. To see how this may be treated, the transfer function of the first-order recursive DFT structure can be noted. Multiplying both the numerator and the denominator of H$_k$(z) by the factor $(1-W_N^k z^{-1})$, we obtain second-order transfer function as

$$H_k[z]=\frac{1-W_N^k z^{-1}}{(1-W_N^{-k}z^{-1})(1-W_N^k z^{-1})}=\frac{1-W_N^k z^{-1}}{1-2\cos(2\pi k/N)z^{-1}+z^{-2}}. \qquad (7)$$

2.1.2 The Recursive DFT Architecture



(a) (b)

Fig. 2: (a) Block diagram of the first-order recursive DFT structure and (b) a multiplexer-type dash-line implementation with down-sampling value of N.

Eq. (6) can be mapped into the first-order recursive DFT structure as shown in Fig. 2(a), where initial rest conditions are assumed and the vertical dash-line denotes the down-sample operation with N for each crossing signal path. Note that the dash-line as shown in Fig. 2(a) can be possibly implemented by multiplexer-type or register-type down-sampling realization. Here, we adopt the multiplexer-type down-sampling realization as shown in Fig. 2(b) due to the advantages of less area and exact mapping from the equation to the architecture. In Fig. 2(b), if *sel*=1, the lower-side signal is passed to the output; otherwise, the upper-side signal is selected as the output signal for the multiplexer. In this correspondence, since the input $x[n]$ and the coefficient W_N^{-k} are in the complex domain, the computation of each new value of $y_k[n]$ through the first-order recursive DFT structure as shown in Fig. 2(a) requires four real multiplications and four real additions. All the intervening values $y_k[1]$, $y_k[2]$,… y_k[N-1] must be computed in order to compute $y_k[N]=X[k]$, so the use of the first-order recursive DFT structure as a computational algorithm requires 4N real multiplications and 4N real additions to compute $X[k]$ for a particular value of k. However, a large number of multiplications are required for the first-order recursive DFT architecture, even if the one avoids the computation or storage of the coefficients W_N^{kn} in Eq. (1) at each nth time index.

Eq. (7) can be mapped into the second-order recursive DFT structure as shown in Fig. 3.

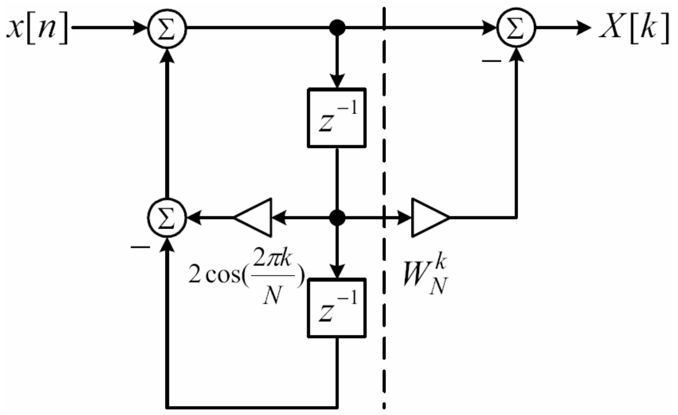


Fig. 3: Block diagram of the second-order recursive DFT structure.

In Fig. 3, only two real multiplications per sample are required to implement the poles of this system as shown in Fig. 3. Note that, in the denominator of Eq. (7), the coefficients are real and the factor –1 need not be counted as a multiplication. It is worthy of emphasizing that the complex multiplication by $-W_N^k$ required to implement the zero of the transfer function need not be performed at every iteration of the difference equation, but only after the Nth iteration. Thus, the total computation is 2N real multiplications and 4N real additions for the poles plus four real multiplications and four real additions for the zero. The coefficients W_N^{kn} are again computed implicitly in the iteration of the recursion formula implied in Fig. 3. The second-order recursive DFT structure can decrease the number of multiplications by Goertzel algorithm; however, the amount of multipliers and the value of the critical period are sacrificed. Hence, the structures in Figs. 2(a) and 3 are not efficient.

2.2 The Review of FFT Algorithm

Due to the large computation load of DFT computation, the direct evaluation of the entire DFT results will cause the serious quantization noise error. FFT are a group of algorithms for significantly speeding up the computation of the DFT. Furthermore, FFT based algorithms reduce the number of computations to achieve the low quantization. Notably, the design with the highest computation complexity also means the highest power consumption [26, 28, 31-33]. The most widely known of these algorithms is attributed to Cooley and Tukey and is used for a number of points N equal to a power-of-two [3]. The number of applications for specific FFTs continues to grow and includes such diverse areas as: speech recognition, video/audio codecs and MIMO-OFDM based mobile communication. There are many ways to measure the complexity and efficiency of an implementation or algorithm, and a final assessment depends on both the available technology the intended application [62]. The arithmetic multiplications and additions are well known to be the good measurements of computational complexity. In this section, some popular FFT algorithms are first reviewed. Some famous pipeline FFT architectures are also detail discussed. Later, some design issues are reminded, such as: high-throughput and long-length based FFT design.

According to the variant of decomposing sequence, two common FFT algorithms could be found, namely decimation in time (DIT) and decimation in frequency (DIF) based FFT algorithms. Significantly, the in-place computation could conveniently make the conversion between these two algorithms [62]. There is no difference in computational complexity and signal flow graph (SFG) between two types of algorithms; herein we only focus on DIF FFT algorithm. In this thesis, we focus on the discussion of DIF based FFT algorithms. Since the low computational complexity of FFT algorithms is desired for high speed and low power consideration in VLSI implementation as discussed before. In this sub-section, the radix-2, radix-4 and radix-8 DIF based equations will be first discussed to demonstrate the computation complexity between different FFT algorithms.

2.2.1 Radix-2 DIF FFT Algorithm

The DIF FFT algorithms are all based on structuring the DFT computation by forming smaller and smaller subsequences of the output sequence $X[k]$. To restrict the formula to N a power of 2, the radix-2 DIF FFT algorithm is to consider computing separately the even-numbered frequency samples and the odd-numbered frequency samples. By separating $X[k]$ into $2r$ and $2r$+1, we obtain the following equations.

$$X[2r] = \sum_{n=0}^{N-1} x[n]\cdot W_N^{n(2r)} = \sum_{n=0}^{\frac{N}{2}-1} x[n]\cdot W_N^{2rn} + \sum_{n=\frac{N}{2}}^{N-1} x[n]\cdot W_N^{2rn} \tag{8}$$

$$X[2r+1] = \sum_{n=0}^{N-1} x[n]\cdot W_N^{n(2r+1)} = \sum_{n=0}^{\frac{N}{2}-1} x[n]\cdot W_N^{(2r+1)n} + \sum_{n=\frac{N}{2}}^{N-1} x[n]\cdot W_N^{(2r+1)n} \tag{9}$$

where r=0,1,……(N/2 - 1). Due to the periodicity of W_N^{2rn}, we could substitute the variables in the second term of summation to obtain the following equations.

$$\begin{aligned} X[2r] &= \sum_{n=0}^{\frac{N}{2}-1} x[n]\cdot W_N^{2rn} + \sum_{n=0}^{\frac{N}{2}-1} x[n+\frac{N}{2}]\cdot W_N^{2r(n+\frac{N}{2})} = \sum_{n=0}^{\frac{N}{2}-1} x[n]\cdot W_N^{2rn} + \sum_{n=0}^{\frac{N}{2}-1} x[n+\frac{N}{2}]\cdot W_N^{2rn} \\ &= \sum_{n=0}^{\frac{N}{2}-1} \{x[n]+x[n+\frac{N}{2}]\} W_N^{2rn} = \sum_{n=0}^{\frac{N}{2}-1} \{x[n]+x[n+\frac{N}{2}]\} W_{\frac{N}{2}}^{rn} \end{aligned} \tag{10}$$

$$\begin{aligned} X[2r+1] &= \sum_{n=0}^{\frac{N}{2}-1} x[n]\cdot W_N^{(2r+1)n} + \sum_{n=0}^{\frac{N}{2}-1} x[n+\frac{N}{2}]\cdot W_N^{(2r+1)(n+\frac{N}{2})} \\ &= \sum_{n=0}^{\frac{N}{2}-1} x[n]\cdot W_N^{(2r+1)n} - \sum_{n=0}^{\frac{N}{2}-1} x[n+\frac{N}{2}]\cdot W_N^{(2r+1)n} \\ &= \sum_{n=0}^{\frac{N}{2}-1} \{x[n]-x[n+\frac{N}{2}]\}\cdot W_N^{(2r+1)n} = \sum_{n=0}^{\frac{N}{2}-1} \{x[n]-x[n+\frac{N}{2}]\}\cdot W_N^{rn} W_{\frac{N}{2}}^{rn} \end{aligned} \tag{11}$$

Following with the similar decomposition procedure, two N/2 points DFT results can be further decomposed and then four N/4 points DFT results are produced. After $\log_2 N$ time recursive decompositions, we can obtain the radix-2 DIF FFT algorithm.

Considering the computation complexity, the direct DFT computation requires a total of

N^2 complex multiplications and $N(N\text{-}1)$ complex additions. It is well known that each complex multiplication requires four real multiplications and two real additions, and each complex addition requires two real additions. Then, the direct computation of DFT of a sequence $x[n]$ totally requires $4N^2$ real multiplications and $N(4N\text{-}2)$ real additions. From the eqs. (10) and (11), the radix-2 algorithm requires $N\log_2 N$ complex multiplications and complex additions. Alternately, the radix-2 algorithm requires $\frac{3N}{2}\log_2 N - \frac{7}{2}N + 8$ real multiplications and $\frac{5N}{2}\log_2 N - \frac{7}{2}N + 8$ real additions.

2.2.2 Radix-4 DIF FFT Algorithm

From the discussion in subsection 2.2.1.1, it is obviously that the radix-2 DIF FFT algorithm could efficiently compute the DFT results than direct method. Comparing with the radix-2 algorithm, the radix-4 algorithm can further reduce the computation complexity with keeping the same regularity in each butterfly computation. A radic-4 DIF FFT algorithm can be derived from recursively decimating the frequency series into four subsets. By separating $X[k]$ into $4r$, $4r$+1, $4r$+2 and $4r$+3, we obtain the following equations.

$$X[4r] = \sum_{n=0}^{N-1} x[n]\cdot W_N^{4rn} = \sum_{n=0}^{\frac{N}{4}-1} x[n]\cdot W_N^{4rn} + \sum_{n=\frac{N}{4}}^{\frac{N}{2}-1} x[n]\cdot W_N^{4rn} + \sum_{n=\frac{N}{2}}^{\frac{3N}{4}-1} x[n]\cdot W_N^{4rn} + \sum_{n=\frac{3N}{4}}^{N-1} x[n]\cdot W_N^{4rn} \tag{12}$$

$$\begin{aligned} X[4r+1] &= \sum_{n=0}^{N-1} x[n]\cdot W_N^{(4r+1)n} \\ &= \sum_{n=0}^{\frac{N}{4}-1} x[n]\cdot W_N^{(4r+1)n} + \sum_{n=\frac{N}{4}}^{\frac{N}{2}-1} x[n]\cdot W_N^{(4r+1)n} + \sum_{n=\frac{N}{2}}^{\frac{3N}{4}-1} x[n]\cdot W_N^{(4r+1)n} + \sum_{n=\frac{3N}{4}}^{N-1} x[n]\cdot W_N^{(4r+1)n} \end{aligned} \tag{13}$$

$$X[4r+2] = \sum_{n=0}^{N-1} x[n]\cdot W_N^{(4r+2)n}$$

$$= \sum_{n=0}^{\frac{N}{4}-1} x[n]\cdot W_N^{(4r+2)n} + \sum_{n=\frac{N}{4}}^{\frac{N}{2}-1} x[n]\cdot W_N^{(4r+2)n} + \sum_{n=\frac{N}{2}}^{\frac{3N}{4}-1} x[n]\cdot W_N^{(4r+2)n} + \sum_{n=\frac{3N}{4}}^{N-1} x[n]\cdot W_N^{(4r+2)n} \tag{14}$$

$$X[4r+3] = \sum_{n=0}^{N-1} x[n]\cdot W_N^{(4r+3)n}$$

$$= \sum_{n=0}^{\frac{N}{4}-1} x[n]\cdot W_N^{(4r+3)n} + \sum_{n=\frac{N}{4}}^{\frac{N}{2}-1} x[n]\cdot W_N^{(4r+3)n} + \sum_{n=\frac{N}{2}}^{\frac{3N}{4}-1} x[n]\cdot W_N^{(4r+3)n} + \sum_{n=\frac{3N}{4}}^{N-1} x[n]\cdot W_N^{(4r+3)n} \tag{15}$$

where r=0,1,……(N/4 - 1). Due to the periodicity of W_N^{4rn}, we could substitute the variables in the 2nd, 3rd and 4th term of summation to obtain the following equations.

$$X[4r] = \sum_{n=0}^{\frac{N}{4}-1} x[n]\cdot W_N^{4rn} + \sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{N}{4}]\cdot W_N^{4r(n+\frac{N}{4})}$$

$$+ \sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{2N}{4}]\cdot W_N^{4r(n+\frac{2N}{4})} + \sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{3N}{4}]\cdot W_N^{4r(n+\frac{3N}{4})}$$

$$= \sum_{n=0}^{\frac{N}{4}-1} \{(x[n]+x[n+\frac{N}{2}]) + (x[n+\frac{N}{4}]+x[n+\frac{3N}{4}])\} W_{\frac{N}{4}}^{rn} \tag{16}$$

$$X[4r+1] = \sum_{n=0}^{\frac{N}{4}-1} x[n]\cdot W_N^{(4r+1)n} + \sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{N}{4}]\cdot W_N^{(4r+1)(n+\frac{N}{4})}$$

$$+ \sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{2N}{4}]\cdot W_N^{(4r+1)(n+\frac{2N}{4})} + \sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{3N}{4}]\cdot W_N^{(4r+1)(n+\frac{3N}{4})}$$

$$= \sum_{n=0}^{\frac{N}{4}-1} \{(x[n]-x[n+\frac{N}{2}]) - j(x[n+\frac{N}{4}]-x[n+\frac{3N}{4}])\} W_N^{n} W_{\frac{N}{4}}^{rn} \tag{17}$$

$$X[4r+2] = \sum_{n=0}^{\frac{N}{4}-1} x[n]\cdot W_N^{(4r+2)n} + \sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{N}{4}]\cdot W_N^{(4r+2)(n+\frac{N}{4})}$$

$$+\sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{2N}{4}]\cdot W_N^{(4r+2)(n+\frac{2N}{4})} + \sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{3N}{4}]\cdot W_N^{(4r+2)(n+\frac{3N}{4})}$$

$$=\sum_{n=0}^{\frac{N}{4}-1}\{(x[n]+x[n+\frac{N}{2}])-(x[n+\frac{N}{4}]+x[n+\frac{3N}{4}])\}W_N^{2n}W_{\frac{N}{4}}^{rn} \tag{18}$$

$$X[4r+3]=\sum_{n=0}^{\frac{N}{4}-1} x[n]\cdot W_N^{(4r+3)n} + \sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{N}{4}]\cdot W_N^{(4r+3)(n+\frac{N}{4})}$$

$$+\sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{2N}{4}]\cdot W_N^{(4r+3)(n+\frac{2N}{4})} + \sum_{n=0}^{\frac{N}{4}-1} x[n+\frac{3N}{4}]\cdot W_N^{(4r+3)(n+\frac{3N}{4})}$$

$$=\sum_{n=0}^{\frac{N}{4}-1}\{(x[n]-x[n+\frac{N}{2}])+j(x[n+\frac{N}{4}]-x[n+\frac{3N}{4}])\}W_N^{3n}W_{\frac{N}{4}}^{rn} \tag{19}$$

Following with the similar decomposition procedure, four $N/4$ points DFT results can be further decomposed and then sixteen $N/16$ points DFT results are produced. After $\log_4 N$ time recursive decompositions, we can obtain the radix-4 DIF FFT algorithm.

Considering the computation complexity, the radix-4 algorithm requires $N\log_4 N$ complex multiplications and complex additions from the eq. (16) and (19). Alternately, the radix-4 algorithm requires $\frac{9N}{8}\log_2 N-3N+3$ real multiplications and $\frac{9N}{8}\log_2 N-3N+3$ real additions.

2.2.3 Radix-8 DIF FFT Algorithm

Following with the similar decomposition produce, a radix-8 DIF FFT algorithm can be derived from recursively decimating the frequency series into eight subsets. After the separation of $X[k]$ into $8r$, $8r$+1, $8r$+2, $8r$+3, $8r$+4, $8r$+5, $8r$+6 and $8r$+7, we adopt the periodicity of W_N^{8rn} to obtain the following equations.

$$X[8r]=\sum_{n=0}^{N-1}x[n]\cdot W_N^{8rn}=\sum_{n=0}^{\frac{N}{8}-1}\{[(x[n]+x[n+\frac{N}{2}])+(x[n+\frac{N}{4}]+x[n+\frac{3N}{4}])]$$
$$+[(x[n+\frac{N}{8}]+x[n+\frac{5N}{8}])+(x[n+\frac{3N}{8}]+x[n+\frac{7N}{8}])]\}W_{\frac{N}{8}}^{rn} \qquad (20)$$

$$X[8r+1]=\sum_{n=0}^{N-1}x[n]\cdot W_N^{(8r+1)n}=\sum_{n=0}^{\frac{N}{8}-1}\{[(x[n]-x[n+\frac{N}{2}])-j(x[n+\frac{N}{4}]-x[n+\frac{3N}{4}])]$$
$$+W_8^1[(x[n+\frac{N}{8}]-x[n+\frac{5N}{8}])-j(x[n+\frac{3N}{8}]-x[n+\frac{7N}{8}])]\}W_{\frac{N}{8}}^{rn}W_N^{n} \qquad (21)$$

$$X[8r+2]=\sum_{n=0}^{N-1}x[n]\cdot W_N^{(8r+2)n}=\sum_{n=0}^{\frac{N}{8}-1}\{[(x[n]+x[n+\frac{N}{2}])-(x[n+\frac{N}{4}]+x[n+\frac{3N}{4}])]$$
$$-j[(x[n+\frac{N}{8}]-x[n+\frac{5N}{8}])+j(x[n+\frac{3N}{8}]-x[n+\frac{7N}{8}])]\}W_{\frac{N}{8}}^{rn}W_N^{2n} \qquad (22)$$

$$X[8r+3]=\sum_{n=0}^{N-1}x[n]\cdot W_N^{(8r+3)n}=\sum_{n=0}^{\frac{N}{8}-1}\{[(x[n]-x[n+\frac{N}{2}])+j(x[n+\frac{N}{4}]+x[n+\frac{3N}{4}])]$$
$$+W_8^3[(x[n+\frac{N}{8}]-x[n+\frac{5N}{8}])+j(x[n+\frac{3N}{8}]-x[n+\frac{7N}{8}])]\}W_{\frac{N}{8}}^{rn}W_N^{3n} \qquad (23)$$

$$X[8r+4]=\sum_{n=0}^{N-1}x[n]\cdot W_N^{(8r+4)n}=\sum_{n=0}^{\frac{N}{8}-1}\{[(x[n]+x[n+\frac{N}{2}])+(x[n+\frac{N}{4}]+x[n+\frac{3N}{4}])]$$
$$-[(x[n+\frac{N}{8}]-x[n+\frac{5N}{8}])+(x[n+\frac{3N}{8}]-x[n+\frac{7N}{8}])]\}W_{\frac{N}{8}}^{rn}W_N^{4n} \qquad (24)$$

$$X[8r+5]=\sum_{n=0}^{N-1}x[n]\cdot W_N^{(8r+5)n}=\sum_{n=0}^{\frac{N}{8}-1}\{[(x[n]-x[n+\frac{N}{2}])-j(x[n+\frac{N}{4}]-x[n+\frac{3N}{4}])]$$

$$-W_8^1[(x[n+\frac{N}{8}]-x[n+\frac{5N}{8}])-j(x[n+\frac{3N}{8}]-x[n+\frac{7N}{8}])]\}W_{\frac{N}{8}}^{rn}W_N^{5n} \quad (25)$$

$$X[8r+6]=\sum_{n=0}^{N-1}x[n]\cdot W_N^{(8r+6)n}=\sum_{n=0}^{\frac{N}{8}-1}\{[(x[n]+x[n+\frac{N}{2}])-(x[n+\frac{N}{4}]-x[n+\frac{3N}{4}])]$$

$$+j[(x[n+\frac{N}{8}]+x[n+\frac{5N}{8}])-(x[n+\frac{3N}{8}]+x[n+\frac{7N}{8}])]\}W_{\frac{N}{8}}^{rn}W_N^{6n} \quad (26)$$

$$X[8r+7]=\sum_{n=0}^{N-1}x[n]\cdot W_N^{(8r+7)n}=\sum_{n=0}^{\frac{N}{8}-1}\{[(x[n]-x[n+\frac{N}{2}])+j(x[n+\frac{N}{4}]-x[n+\frac{3N}{4}])]$$

$$-W_8^3[(x[n+\frac{N}{8}]-x[n+\frac{5N}{8}])+j(x[n+\frac{3N}{8}]-x[n+\frac{7N}{8}])]\}W_{\frac{N}{8}}^{rn}W_N^{7n} \quad (27)$$

where r=0,1,……(N/8 - 1). Following with the similar decomposition procedure, eight N/8 points DFT results can be further decomposed and then 64 N/64 points DFT results are produced. After $\log_8 N$ time recursive decompositions, we can obtain the radix-8 DIF FFT algorithm.

Considering the computation complexity, the radix-8 algorithm requires $N\log_8 N$ complex multiplications and complex additions from the eq. (20) to (17). Alternately, the radix-8 algorithm requires $\frac{21N}{24}\log_2 N-\frac{25N}{8}+4$ real multiplications and $\frac{8T+73N}{24}\log_2 N-\frac{25N}{8}+4$ real additions, where T denotes the number of real additions for the realization of coefficient W_8^l.

2.2.4 Radix-2/4 DIF FFT Algorithm

Duhamel *et al.* [63] presented the radix-2/4 and radix-2/8 FFT algorithms, which achieve the few multiplications and additions. The radix-2/4 algorithm takes the advantages of both radix-2 and radix-4 algorithms. On the other hand, the radix-2/8 has the advantages of both radix-2 and radix-8 algorithms. However, the radix-2/4 and radix-2/8 algorithms are less regular than the fixed-radix based algorithms. To decimate the frequency series into even-numbered points and odd-numbered points, the radix-2/4 DIF FFT algorithm can be obtained. After the separation of $X[k]$ into $2r$, and $2r+1$, we adopt the radix-4 decomposition with the periodicity of W_N^{2rn} and W_N^{4sn} to obtain the following equations.

$$X[2r]=\sum_{n=0}^{N-1}x[n]\cdot W_N^{n(2r)}=\sum_{n=0}^{\frac{N}{2}-1}\{x[n]+x[n+\frac{N}{2}]\}W_{\frac{N}{2}}^{rn} \tag{28}$$

$$\begin{aligned}X[4s+1]&=\sum_{n=0}^{\frac{N}{4}-1}x[n]\cdot W_N^{(4s+1)n}+\sum_{n=0}^{\frac{N}{4}-1}x[n+\frac{N}{4}]\cdot W_N^{(4s+1)(n+\frac{N}{4})}\\&+\sum_{n=0}^{\frac{N}{4}-1}x[n+\frac{2N}{4}]\cdot W_N^{(4s+1)(n+\frac{2N}{4})}+\sum_{n=0}^{\frac{N}{4}-1}x[n+\frac{3N}{4}]\cdot W_N^{(4s+1)(n+\frac{3N}{4})}\\&=\sum_{n=0}^{\frac{N}{4}-1}\{(x[n]-x[n+\frac{N}{2}])-j(x[n+\frac{N}{4}]-x[n+\frac{3N}{4}])\}W_N^{n}W_{\frac{N}{4}}^{sn}\end{aligned} \tag{29}$$

$$\begin{aligned}X[4s+3]&=\sum_{n=0}^{\frac{N}{4}-1}x[n]\cdot W_N^{(4s+3)n}+\sum_{n=0}^{\frac{N}{4}-1}x[n+\frac{N}{4}]\cdot W_N^{(4s+3)(n+\frac{N}{4})}\\&+\sum_{n=0}^{\frac{N}{4}-1}x[n+\frac{2N}{4}]\cdot W_N^{(4s+3)(n+\frac{2N}{4})}+\sum_{n=0}^{\frac{N}{4}-1}x[n+\frac{3N}{4}]\cdot W_N^{(4s+3)(n+\frac{3N}{4})}\\&=\sum_{n=0}^{\frac{N}{4}-1}\{(x[n]-x[n+\frac{N}{2}])+j(x[n+\frac{N}{4}]-x[n+\frac{3N}{4}])\}W_N^{3n}W_{\frac{N}{4}}^{sn}\end{aligned} \tag{30}$$

where r=0,1,……(N/2 - 1) and s=0,1,……(N/4 – 1) . Considering the computation complexity, the radix-2/4 algorithm requires $N\log_2 N-3N+4$ real multiplications and $3N\log_2 N-3N+4$ real additions.

2.2.5 Radix-2/8 DIF FFT Algorithm

Following the similar decomposition produce with radix-2/4 algorithm, the radix-2/8 algorithm can be derived by recursively decimating the frequency series into even-numbered points and odd-numbered points. After the separation of $X[k]$ into $2r$, and $2r+1$, we adopt the radix-8 decomposition with the periodicity of W_N^{2rn} and W_N^{8sn} to obtain the following equations.

$$X[2r] = \sum_{n=0}^{N-1} x[n] \cdot W_N^{n(2r)} = \sum_{n=0}^{\frac{N}{2}-1} \{x[n] + x[n+\frac{N}{2}]\} W_{\frac{N}{2}}^{rn} \tag{31}$$

$$\begin{aligned} X[8s+1] = \sum_{n=0}^{N-1} x[n] \cdot W_N^{(8s+1)n} &= \sum_{n=0}^{\frac{N}{8}-1} \{[(x[n] - x[n+\frac{N}{2}]) - j(x[n+\frac{N}{4}] - x[n+\frac{3N}{4}])] \\ &+ W_8^1[(x[n+\frac{N}{8}] - x[n+\frac{5N}{8}]) - j(x[n+\frac{3N}{8}] - x[n+\frac{7N}{8}])]\} W_{\frac{N}{8}}^{sn} W_N^{n} \end{aligned} \tag{32}$$

$$\begin{aligned} X[8s+3] = \sum_{n=0}^{N-1} x[n] \cdot W_N^{(8s+3)n} &= \sum_{n=0}^{\frac{N}{8}-1} \{[(x[n] - x[n+\frac{N}{2}]) + j(x[n+\frac{N}{4}] + x[n+\frac{3N}{4}])] \\ &+ W_8^3[(x[n+\frac{N}{8}] - x[n+\frac{5N}{8}]) + j(x[n+\frac{3N}{8}] - x[n+\frac{7N}{8}])]\} W_{\frac{N}{8}}^{sn} W_N^{3n} \end{aligned} \tag{33}$$

$$\begin{aligned} X[8s+5] = \sum_{n=0}^{N-1} x[n] \cdot W_N^{(8s+5)n} &= \sum_{n=0}^{\frac{N}{8}-1} \{[(x[n] - x[n+\frac{N}{2}]) - j(x[n+\frac{N}{4}] - x[n+\frac{3N}{4}])] \\ &- W_8^1[(x[n+\frac{N}{8}] - x[n+\frac{5N}{8}]) - j(x[n+\frac{3N}{8}] - x[n+\frac{7N}{8}])]\} W_{\frac{N}{8}}^{sn} W_N^{5n} \end{aligned} \tag{34}$$

$$\begin{aligned} X[8s+7] = \sum_{n=0}^{N-1} x[n] \cdot W_N^{(8s+7)n} &= \sum_{n=0}^{\frac{N}{8}-1} \{[(x[n] - x[n+\frac{N}{2}]) + j(x[n+\frac{N}{4}] - x[n+\frac{3N}{4}])] \\ &- W_8^3[(x[n+\frac{N}{8}] - x[n+\frac{5N}{8}]) + j(x[n+\frac{3N}{8}] - x[n+\frac{7N}{8}])]\} W_{\frac{N}{8}}^{sn} W_N^{7n} \end{aligned} \tag{35}$$

where r=0,1,……(N/2 - 1) and s=0,1,……(N/8 – 1) . Considering the computation complexity, the radix-2/8 algorithm requires $M_R(N)$ real multiplications and $M_R(N)+2N\log_2 N$ real additions. The notation of $M_R(N)$ could be defined as

$$M_R(N) = (n-2.75)N + 3 + [9(\sqrt{2})^{n-5} - \frac{N}{4}(\frac{1}{\sqrt{2}})^{n-5}]\cos(n-5)\theta$$
$$-\frac{N}{4}(\frac{1}{\sqrt{2}})^{n-5}]\frac{1}{\sqrt{7}}\sin(n-5)\theta \qquad (36)$$

2.2.6 Radix-2^2 DIF FFT Algorithm

He *et al.* [31] presented the radix-2^2 and radix-2^3 FFT algorithms, which achieve the low computational complexities of the radix-4 and radix-8 algorithms but the low hardware requirement of the radix-2 algorithm. The radix-2^2 algorithm takes the advantages of both radix-2 and radix-4 algorithms. On the other hand, the radix-2^3 has the advantages of both radix-2 and radix-8 algorithms. Furthermore, the radix-2^2 and radix-2^3 algorithms keep the regularity with the fixed-radix based algorithms. Applying a 3-dimensional linear index map, the parameters n and k of eq. (1) could be expressed as the combinations of n_1, n_2, n_3 and k_1, k_2, k_3, respectively.

$$n = \frac{N}{2}n_1 + \frac{N}{4}n_2 + n_3, \quad k = k_1 + 2k_2 + 4k_3. \qquad (37)$$

where $0 \leqq n_1, n_2, k_1, k_2 \leqq 1$.The common factor algorithm (CFA) [64] form can be written as

$$X[k_1 + 2k_2 + 4k_3] = \sum_{n_3=0}^{\frac{N}{4}-1}\sum_{n_2=0}^{1}\sum_{n_1=0}^{1} x(\frac{N}{2}n_1 + \frac{N}{4}n_2 + n_3)W_N^{(\frac{N}{2}n_1+\frac{N}{4}n_2+n_3)(k_1+2k_2+4k_3)}$$
$$= \left\{\sum_{n_3=0}^{\frac{N}{4}-1}\left\{\sum_{n_2=0}^{1} B_{\frac{N}{2}}^{k_1}(\frac{N}{4}n_2 + n_3)W_N^{(\frac{N}{4}n_2+n_3)k_1}\right\}W_N^{(\frac{N}{4}n_2+n_3)(2k_2+4k_3)}\right\}, \qquad (38)$$

where the butterfly structure of the first stage takes the form

$$B_{\frac{N}{2}}^{k_1}(\frac{N}{4}n_2 + n_3) = x(\frac{N}{4}n_2 + n_3) + (-1)^{k_1}x(\frac{N}{4}n_2 + n_3 + \frac{N}{2}), \qquad (39)$$

Following a similar decomposition procedure, Eq. (38) can be decomposed as

$$X[k_1+2k_2+4k_3] = \left\{ \sum_{n_3=0}^{\frac{N}{4}-1} B_{\frac{N}{4}}^{k_1,k_2}(n_3) W_N^{n_3(k_1+2k_2)} \right\} W_{\frac{N}{4}}^{n_3 k_3}, \tag{40}$$

Meanwhile, the butterfly structure of the second stage can be obtained as

$$B_{\frac{N}{4}}^{k_1,k_2}(n_3) = B_{\frac{N}{2}}^{k_1}(n_3) + (-j)^{(k_1+2k_2)} B_{\frac{N}{2}}^{k_1}(n_3+\frac{N}{4}), \tag{41}$$

From eqs (38) and (40), it is obviously that the radix-2^2 algorithm reduces the non-trivial multiplications as the radix-4 algorithm. Furthermore, the radix-2^2 algorithm still keeps the radix-2 butterfly structure as depicted in (39) and (41).

2.2.7 Radix-2^3 DIF FFT Algorithm

Applying another 4-dimensional linear index map in (1), the parameters n and k could be expressed as the combinations of n_1, n_2, n_3, n_4 and k_1, k_2, k_3, k_4 , respectively.

$$n = \frac{N}{2}n_1 + \frac{N}{4}n_2 + \frac{N}{16}n_3 + n_4,$$

$$k = k_1 + 2k_2 + 4k_3 + 16k_4 . \tag{42}$$

where $0 \leqq n_1, n_2, n_3, k_1, k_2, k_3 \leqq 1$. The common factor algorithm (CFA) [64] form can be written as

$$X[k_1+2k_2+4k_3+16k_4] = \sum_{n_4=0}^{\frac{N}{16}-1} \Bigg\{ \sum_{n_3=0}^{1} \Bigg\{ \sum_{n_2=0}^{1} B_{\frac{N}{2}}^{k_1}(\frac{N}{4}n_2 + \frac{N}{8}n_3 + n_4)$$

$$\cdot W_N^{\frac{N}{4}n_2(k_1+2k_2)} \Bigg\} W_N^{\frac{N}{8}n_3(k_1+2k_2+4k_3)} \Bigg\} W_N^{n_4(k_1+2k_2+4k_3+8k_4)}$$

$$= \sum_{n_4=0}^{\frac{N}{8}-1} \left\{ \sum_{n_3=0}^{1} B_{\frac{N}{4}}^{k_1,k_2}(\frac{N}{8}n_3 + n_4) W_N^{\frac{N}{8}n_3(k_1+2k_2+4k_3)} \right\} \cdot W_N^{n_4(k_1+2k_2+4k_3+16k_4)}$$

$$= \sum_{n_4=0}^{\frac{N}{8}-1} \left\{ B_{\frac{N}{8}}^{k_1,k_2,k_3}(n_4) \cdot W_N^{n_4(k_1+2k_2+4k_3)} \right\} \cdot W_{\frac{N}{8}}^{n_4 k_4}, \tag{43}$$

where the butterfly structure of the each stage takes the form

$$B_{\frac{N}{2}}^{k_1}(\frac{N}{4}n_2+n_3)=x(\frac{N}{4}n_2+n_3)+(-1)^{k_1}x(\frac{N}{4}n_2+n_3+\frac{N}{2})$$

$$B_{\frac{N}{4}}^{k_1,k_2}(n_3)=B_{\frac{N}{2}}^{k_1}(n_3)+(-j)^{(k_1+2k_2)}B_{\frac{N}{2}}^{k_1}(n_3+\frac{N}{4})$$

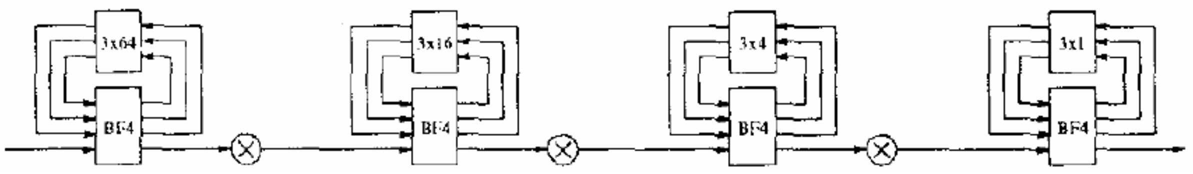
$$B_{\frac{N}{8}}^{k_1,k_2,k_3}(n_4)=B_{\frac{N}{4}}^{k_1,k_2}(n_4)+W_N^{\frac{N}{8}(k_1+2k_2+4k_3)}B_{\frac{N}{4}}^{k_1,k_2}(n_4+\frac{N}{8})\,, \tag{44}$$

From eqs (43), it is obviously that the radix-2^3 algorithm reduces the non-trivial multiplications as the radix-8 algorithm. Furthermore, the radix-2^3 algorithm still keeps the radix-4 butterfly structure as depicted in (44).

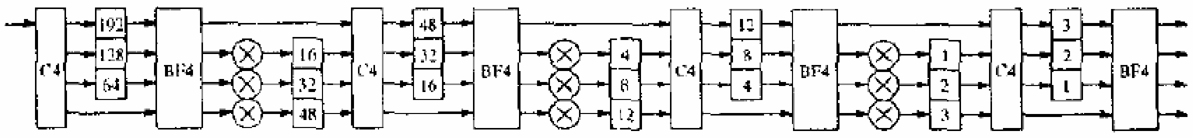
In 1998, He and Torkeson suggested radix-2^2 FFT algorithm [31]. The reason to develop a radix-2^2 algorithm instead of conventional radix-4 and radix-2 FFT is that the number of the non-trivial multiplications can be further reduced in implementation. The radix-2^2 algorithm is characterized with the same multiplication complexity as the radix-4 algorithm but still retain the radix-2 butterfly structure. A radix-2^2 DIF FFT algorithm can be derived by recursively decimating the frequency series into four subsets. By substituting k for $4r+2s_2+s_1$, it follows from equation (1) that

2.3 The Review of Pipeline FFT Architecture

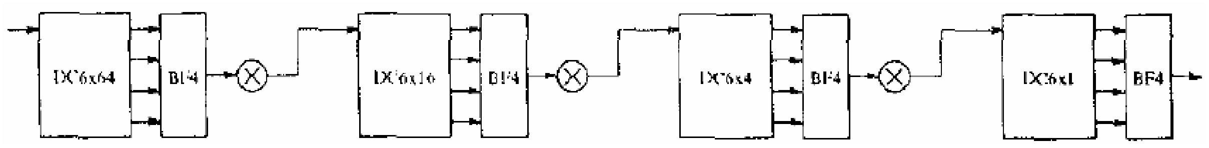
Many researches were concentrated on the design of efficient FFT architecture. For the purpose of achieving the most effective architecture, the appropriated algorithm and architecture for the FFT processor should be chosen trading off its processing speed and its chip cost. We could use five performance measures to define the efficiency of related FFT architectures, which includes: input data organization, output data organization, internal data bus loading, throughput and computation latency [1]. There are two types of data buffering structures for pipelined-based FFT architecture, that are delay-commutator (DC) and delay-feedback (DF). Base on these two structures, three different pipeline architectures could be found: single-path delay feedback (SDF), multiple-path delay commutator (MDC) and single-path delay commutator (SDC) architecture. Base on these three pipeline architectures, figure 4 lists the radix-4 based 256-points pipeline FFT processors. According to the five measures, the SDF architecture is well known to be more efficient than MDC and SDC architectures in terms of input data ordering, output data ordering and internal bus loading. Due to memory sharing in SDF architecture, the butterfly output uses the same storage with its input. Although, the MDC architecture has the higher throughput rate than SDF architecture, the MDC architecture spends the larger chip cost than SDF architecture. In Fig. 4, the R4MDC architecture has four times throughput rate than R4SDF architecture. However, the R4MDC architecture also increases 3 and 1.7 times of complex multipliers and memories in the 256-points FFT computation. The most effective FFT processor should consider the tightly hardware scheduling and chip cost at the same time.



(a) The R4SDF architecture.



(b) The R4MDC architecture.



(c) The R4SDC architecture.

Fig. 4: Three 256-points pipeline FFT architecture.

2.4 The MIMO-FFT Architecture

The High Throughput Task Group, which established the IEEE 802.11n standard, is going to develop the next-generation wireless LAN (WLAN) based on the 802.11 a/g, which comprises the current OFDM-based WLAN standards [23]. According to the IEEE 802.11n standard [23], 128-point and 64-point FFT/IFFT processors are utilized to support four different throughput rates —R, 2R, 3R and 4R—within 3.6 or 4 μs. The transmitted signal bandwidths are 40 and 20 MHz for the 128-point and 64-point FFT/IFFT processors, respectively. In this study, we focus our 64-point FFT/IFFT design on 2×2 and 4×4 MIMO-OFDM WLAN systems, which require the high throughput rate of 2R and 4R. Sansaloni *et al.* presented a detail comparison of several 64-points FFT/IFFT algorithms for the MIMO-OFDM WLAN system [26]. According to that comparison, the multi-path delay commutator (MDC) based design, which was built by the serial blockwise architecture, is the most cost-efficient architecture for the MIMO-OFDM system. For a 4×4 MIMO-OFDM system, the radix-4 multi-path delay commutator (R4MDC) architecture can achieve the lowest hardware requirement, where the operating frequency equals the sampling frequency, while the radix-2 multi-path delay commutator (R2MDC) architecture is the most cost-efficient architecture for the 2×2 MIMO-OFDM system. However, the R4MDC- and

R2MDC- based 64-point FFT/IFFT designs both have higher complex multiplicative complexities than the radix-8, radix-2/4/8 and radix-2/8 based designs as listed in Table 1. The design with the highest complex multiplicative complexity has the highest power consumption [26, 31, 32, 36, 56, 57]. Maharatna *et al.* [41] recently presented a modified radix-8 multi-path delay commutator (R8MDC) based 64-point FFT/IFFT WLAN processor to reduce the hardware cost than the conventional R8MDC design with the appropriate throughput rate of 5.33R. Although, the modified R8MDC design achieves the low complex multiplicative complexity as radix-8 based algorithm, the large amount of memory and four constant multipliers still lead to a large chip cost.

Table 1: Number of complex multiplication needed for the computation of a 64 point FFT/IFFT processor.

	Complex Multiplication	Constant Multiplication
Radix-2	98	N/A
Radix-2^2	76	N/A
Radix-4	76	N/A
Radix-2/4	72	N/A
Radix-2/4/8	48	32
Radix-8	48	32
Radix-2/8	48	32

Bouguezel *et al.* [59] reported the comprehensive analysis of the data transfer, address generation and twiddle factor evaluation or access to the lookup table. The comparison results of [59] reveal that the radix-2/8 algorithm has fewer arithmetic operations than other low-radix and mixed-radix algorithms. Additional, Yeh *et al.* [32] indicate that the radix-2/8 algorithm is computationally superior to all other algorithms, since it has most trivial multiplications (i.e., ± 1 and $\pm j$). Therefore, the radix-2/8 based architecture is presented for the few constant multipliers, high utilization and low complex multiplicative complexity. Yeh *et al.* [32] apply the radix-2/8 algorithm to present the radix-2/8 single path delay feedback (R28SDF) -based 64-point FFT/IFFT processors. However, the single path delay feedback (SDF) based architecture [32] has the lowest throughput rate of R. This investigation adopts the novel radix-2/8 algorithm, which is different from the conventional radix-2/8 algorithm [32, 59, 60], to further reduce the constant multiplier requirement in the proposed retrenched 8-point FFT (R8-FFT) unit. Lin *et al.* briefly described the algorithm that is adopted in the SISO-OFDM application [57]. This work adopts this novel radix-2/8 algorithm and the multiplier after write (MAW) scheme [57] to devise two architectures, radix-2/8 multiple-path delay feedback (R28MDF) and radix-2/8 multiple-path delay commutator (R28MDC), for the high throughput rate system of 2R and 4R, respectively.

Chapter 3 The Low-Computation Cycle and Power-Efficient Recursive DFT/IDFT Design

In this chapter, we focus on the design of low-computation cycle and power-efficient recursive DFT/IDFT design. The detailed descriptions of a high-performance VLSI algorithm and architecture by the hybrid of input strength reduction scheme, Chebyshev polynomial, and register-splitting scheme for the DTMF application have been fully provided. The derived algorithm and devised architecture [23] possesses the following features: low-computation cycle (i.e., high throughput) and power efficiency at the expense of slightly increased area overhead compared with the existing recursive DFT/IDFT structures. This chapter is organized as follows. A new recursive DFT/IDFT algorithm and architecture by the hybrid of input strength reduction, Chebyshev polynomial, and register-splitting schemes is revealed in Section 3.1. In Section 3.2, the DTMF application using this new architecture has been demonstrated. After the bit-level SNR simulation, the 212/106-point DFT/IDFT chip has been successfully implemented for the DTMF detector system. In Section 3.3, the comparison results are tabulated in terms of the amount of computation cycles for each output as well as *N*-point DFT/IDFT, the maximum number of the channel density, the clock period, and the number of real multipliers. At last, the concise statements conclude this chapter in Section 3.4.

3.1 New Recursive Algorithm and Architecture

The DFT of the *N*-point input $x[n]$ is defined as

$$y[k]=\sum_{n=0}^{N-1}x[n]\cdot W_N^{kn}=\sum_{n=0}^{N/2-1}x[n]\cdot W_N^{kn}+\sum_{n=N/2}^{N-1}x[n]\cdot W_N^{kn}, \qquad (45)$$

where $W_N=e^{-j2\pi/N}$. By reducing the input strength of the DFT algorithm, equation (45) can be folded as

$$\begin{aligned} y[k] &= \sum_{n=0}^{N/2-1}x'[n]\cdot W_N^{kn}+\sum_{n=0}^{N/2-1}x'[N-1-n]\cdot W_N^{k(N-1-n)} \\ &= \sum_{n=0}^{N/2-1}\left(x'[n]+W_N^{-k}\cdot x'[N-1-n]\right)\cdot\cos\left(\frac{2\pi kn}{N}\right) \end{aligned}$$

$$+ j \sum_{n=0}^{N/2-1} \left(-x'[n] + W_N^{-k} \cdot x'[N-1-n] \right) \cdot \sin\left(\frac{2\pi kn}{N}\right) \quad , \qquad (46)$$

where $x'[n] = \begin{cases} x[n], & 0 \le n \le N/2-1 \\ 0, & \text{otherwise} \end{cases}$. Since using the input strength reduction scheme in (46), only half summation terms are needed to express *y*[*k*]. Equation (46) can be treated as DCT and DST parts, $y_{DCT}[k]$ and $y_{DST}[k]$, respectively, as

$$y_{DCT}[k] = \sum_{n=0}^{N/2-1} \left(x'[n] + W_N^{-k} \cdot x'[N-1-n] \right) \cdot \cos\left(\frac{2\pi kn}{N}\right), \qquad (47)$$

and

$$y_{DST}[k] = -\sum_{n=0}^{N/2-1} \left(x'[n] - W_N^{-k} \cdot x'[N-1-n] \right) \cdot \sin\left(\frac{2\pi kn}{N}\right). \qquad (48)$$

In (47), we can define $r_k[n] = x'[n] + W_N^{-k} \cdot x'[N-1-n]$. Replacing *n* by *N*/2-1-*n*, equation (47) can be rewritten as

$$y_{DCT}[k] = \sum_{n=0}^{N/2-1} r_k[n] \cdot \cos\left(\frac{2\pi kn}{N}\right) = \sum_{n=0}^{N/2-1} r_k[N/2-1-n] \cdot \cos\left(\frac{2\pi k(N/2-1-n)}{N}\right)$$

$$= (-1)^k \sum_{n=0}^{N/2-1} r_k[N/2-1-n] \cdot \cos\left(\frac{2\pi k(n+1)}{N}\right) = (-1)^k \cdot g_{N/2-1}(k), \qquad (49)$$

where $g_{N/2-1}(k) = \sum_{n=0}^{N/2-1} r_k[N/2-1-n] \cdot \cos\left(\frac{2\pi k(n+1)}{N}\right)$. Let $\theta_k = \frac{2\pi k}{N}$, and $g_{N/2-1}(k)$ can be generalized as

$$g_i(k) = \sum_{n=0}^{i} r_k[i-n] \cdot \cos((n+1)\theta_k), \text{ where } i = {N \over 2} - 1 \qquad (50)$$

It is known that Chebyshev polynomials are well defined as

$$\cos(r\theta) = 2\cos((r-1)\theta) \cdot \cos\theta - \cos((r-2)\theta), \qquad (51)$$

$$\sin(r\theta) = 2\sin((r-1)\theta) \cdot \cos\theta - \sin((r-2)\theta). \qquad (52)$$

Using the recursive identity stated in (51), equation (50) can be deduced as

$$g_i(k) = \sum_{n=0}^{i} r_k[i-n] . \cos((n+1)\theta_k) = \sum_{n=0}^{i} r_k[i-n] \cdot \{2\cos(n\theta_k) \cdot \cos\theta_k - \cos((n-1)\theta_k)\}$$

$$= 2 \sum_{n=0}^{i} r_k[i-n] \cdot \cos(n\theta_k) \cdot \cos\theta_k - \sum_{n=0}^{i} r_k[i-n] \cdot \cos((n-1)\theta_k)$$

$$= 2r_k[i]\cdot\cos\theta_k + 2\sum_{n=0}^{i-1} r_k[i-1-n]\cdot\cos((n+1)\theta_k)\cdot\cos\theta_k$$

$$-r_k[i]\cdot\cos\theta_k - r_k[i-1] - \sum_{n=0}^{i-2} r_k[i-2-n]\cdot\cos((n+1)\theta_k)$$

$$= r_k[i]\cdot\cos\theta_k - r_k[i-1] + 2\cos\theta_k\cdot g_{i-1}(k) - g_{i-2}(k). \tag{53}$$

The z-transform of (53) can be denoted as

$$\frac{g(k,z)}{r_k(z)} = \frac{\cos\theta_k - z^{-1}}{1-2\cos\theta_k z^{-1} + z^{-2}}. \tag{54}$$

For the DST part in (48), by letting $s_k[n] = x'[n] - W_N^{-k}\cdot x'[N-1-n]$ and replacing n by $N/2-1-n$, $y_{DST}[k]$ can be derived as

$$y_{DST}[k] = -\sum_{n=0}^{N/2-1} s_k[n]\cdot\sin(\frac{2\pi kn}{N}) = (-1)^k\cdot h_{N/2-1}(k), \tag{55}$$

where $h_{N/2-1}(k) = \sum_{n=0}^{N/2-1} s_k[N/2-1-n]\cdot\sin((n+1)\theta_k)$. Applying recursive identity of (52), $h_{N/2-1}(k)$ can be generalized as

$$h_j(k) = \sum_{n=0}^{j} s_k[j-n]\cdot\sin((n+1)\theta_k)$$

$$= 2\sum_{n=0}^{j-1} s_k[j-1-n]\cdot\sin((n+1)\theta_k)\cdot\cos\theta_k + s_k[j]\cdot\sin\theta_k - \sum_{n=0}^{j-2} s_k[j-2-n]\cdot\sin((n+1)\theta_k)$$

$$= s_k[j]\cdot\sin\theta_k + 2\cos\theta_k\cdot h_{j-1}(k) - h_{j-2}(k). \tag{56}$$

The *z*-transform of (56) can be denoted as

$$\frac{h(k,z)}{s_k(z)} = \frac{\sin\theta_k}{1-2\cos\theta_k z^{-1} + z^{-2}}. \tag{57}$$

Equations (54) and (57) can be easily mapped into the recursive DFT structures as shown in Fig. 5(a) and (b), respectively. Compared with the conventional architectures [51, 52, 62], it is clear that by using the proposed DFT algorithm and architecture can reduce computations cycles by 50%. In other words, with respect to the algorithm derivation, the throughput rate can be easily doubled without increasing the operating frequency.

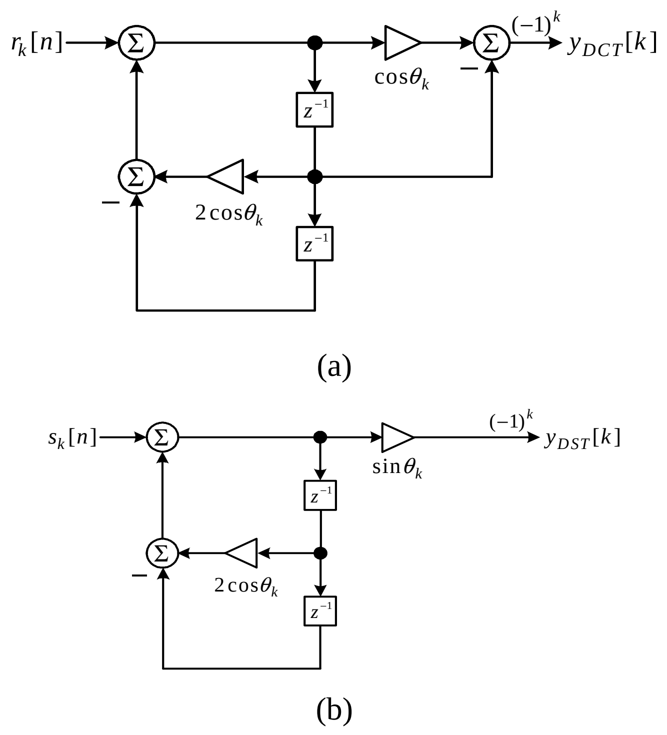


Fig. 5: Block diagram of low-computation cycle for (a) DCT part and (b) DST part of the DFT computation.

For the power-efficiency issue, we adopt the register-splitting scheme [51] (i.e., a type of retiming schemes) to reduce the critical path. There are two main advantages of using retiming scheme [65]: one is high speed and the other is low power. In this paper, we consider this technique for lowering the power consumption where the speed does not need to be increased. The resulting DCT part is depicted in the upper diagram of Fig. 6, where $\boxed{1<=}$ denotes a hardwired shifter with one-bit left shift. Similarly, the DST part can be modified as the lower diagram of Fig. 6. In order to maintain the minimum clock period for the recursive DFT computation, the forward pipeline register, ▨, is exploited for the final sum output. Later combining these two new parts into one, a novel recursive DFT architecture that possesses lower computation cycle and more power-efficiency than the conventional DFT structures can be obtained.

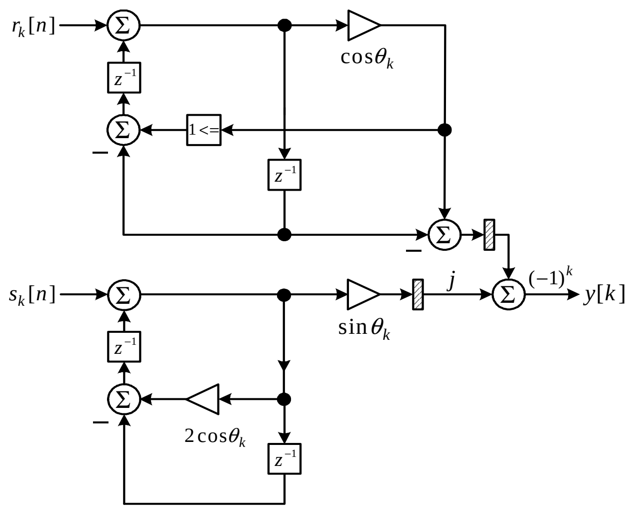


Fig. 6: Block diagram of the proposed low-computation cycle and power-efficiency recursive DFT architecture.

The IDFT of the *N*-point input *y*[*k*] is defined as

$$x[n]=\frac{1}{N}\sum_{k=0}^{N-1} y[k]\cdot W_N^{-kn}, \tag{58}$$

To develop the low-computation cycle recursive IDFT algorithm, equation (58) using the input strength reduction scheme can be modified as

$$x[n]=\frac{1}{N}\sum_{k=0}^{N/2-1}\left(y'[k]+W_N^n\cdot y'[N-1-k]\right)\cdot\cos\left(\frac{2\pi kn}{N}\right)$$
$$+j\cdot\frac{1}{N}\sum_{k=0}^{N/2-1}\left(y'[k]-W_N^n\cdot y'[N-1-k]\right)\cdot\sin\left(\frac{2\pi kn}{N}\right), \tag{59}$$

where $y'[k]=\begin{cases} y[k], & 0\le n\le N/2-1 \\ 0, & \text{otherwise}\end{cases}$. Similarly, equation (59) can be treated as the IDCT and IDST parts, $x_{IDCT}[n]$ and $x_{IDST}[n]$, respectively, as

$$x_{IDCT}[n]=\frac{1}{N}\sum_{k=0}^{N/2-1}\left(y'[k]+W_N^n\cdot y'[N-1-k]\right)\cdot\cos\left(\frac{2\pi kn}{N}\right), \tag{60}$$

$$x_{IDST}[n]=\frac{1}{N}\sum_{k=0}^{N/2-1}\left(y'[k]-W_N^n\cdot y'[N-1-k]\right)\cdot\sin\left(\frac{2\pi kn}{N}\right). \tag{61}$$

In (60), we can define $r_n[k]=y'[k]+W_N^n\cdot y'[N-1-k]$. Replacing *k* by *N*/2-1-*k*, equation (60) can be rewritten as

$$x_{IDCT}[n]=\frac{1}{N}\sum_{k=0}^{N/2-1} r_n[k]\cdot\cos\left(\frac{2\pi kn}{N}\right)=\frac{(-1)^n}{N}\cdot g_{N/2-1}(n), \tag{62}$$

where $g_{N/2-1}(n) = \sum_{k=0}^{N/2-1} r_n[N/2-1-k] \cdot \cos(\frac{2\pi n(k+1)}{N})$. Let $\theta_n = \frac{2\pi n}{N}$, and $g_{N/2-1}(n)$ can be generalized as

$$g_i(n) = \sum_{k=0}^{i} r_n[i-k] \cdot \cos((k+1)\theta_n). \tag{63}$$

Using the recursive identity stated in (51), equation (63) can be deduced as

$$g_i(n) = \sum_{k=0}^{i} r_n[i-k] \cdot \cos((k+1)\theta_n) = r_n[i] \cdot \cos\theta_n - r_n[i-1] + 2\cos\theta_n \cdot g_{i-1}(n) - g_{i-2}(n), \tag{64}$$

The z-transform of (64) can be denoted as

$$\frac{g(n,z)}{r_n(z)} = \frac{\cos\theta_n - z^{-1}}{1 - 2\cos\theta_n z^{-1} + z^{-2}}. \tag{65}$$

For the IDST part in (61), by letting $s_n[k] = y'[k] - W_N^n \cdot y'[N-1-k]$ and replacing k by $N/2-1-k$, $x_{IDST}[n]$ can be derived in similar behavior as

$$x_{IDST}[n] = \frac{1}{N} \sum_{k=0}^{N/2-1} s_n[k] \cdot \sin(\frac{2\pi kn}{N}) = \frac{-(-1)^n}{N} \cdot h_{N/2-1}(n), \tag{66}$$

where $h_{N/2-1}(n) = \sum_{k=0}^{N/2-1} s_n[N/2-1-k] \cdot \sin((k+1)\theta_n)$. Applying (52), $h_{N/2-1}(n)$ can be generalized as

$$h_j(n) = \sum_{k=0}^{j} s_n[j-k] \cdot \sin((k+1)\theta_n) = s_n[j] \cdot \sin\theta_n + 2\cos\theta_n \cdot h_{j-1}(n) - h_{j-2}(n). \tag{67}$$

The *z*-transform of (67) can be denoted as

$$\frac{h(n,z)}{s_n(z)} = \frac{\sin\theta_n}{1 - 2\cos\theta_n z^{-1} + z^{-2}}. \tag{68}$$

After using the register-splitting scheme, equations (65) and (68) can be easily mapped into the modified structures as shown in Fig. 7. Again, from the proposed algorithm and architecture, it is obviously found that the 50% computation cycle reduction can be achieved by contrast with that of [50, 51, 62]. That means double the throughput rate can be achieved under the same operating frequency.

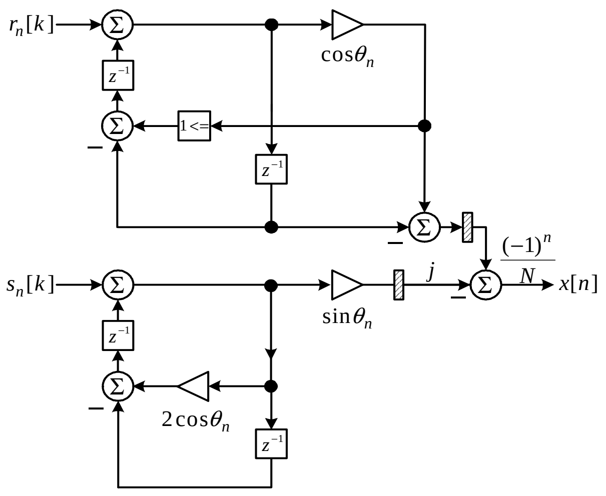


Fig. 7: Block diagram of the proposed low-computation cycle and power-efficient recursive IDFT architecture.

3.2 The Proposed DTMF Receiver and Chip Implementation

In this chapter, we are encouraged to design a low-computation cycle (i.e., high throughput) and power-efficient (i.e., cost-effective) recursive DFT/IDFT architecture for the high channel density DTMF detector in the VoP application. So as to reach this purpose, we follow two down-to-earth steps to optimize our target design. First, according to the dataflow of the DTMF detection as shown in Fig. 8 [13], we could find that the DTMF detector enables one channel telephone [13] to provide 14 different recursive DFT computations. The total computations for the DTMF detector include 6 106-sample frames and 8 212-sample frames. Thus, we proposed one high channel density DTMF detector to handle both 212 and 106-sample frames based on the proposed recursive core architecture as shown in Fig. 9. The proposed architecture in the first 106-sample frame needs full 106 clock cycles because it involves extra 53 clock cycles for the input data latency. The other 5 106-sample frames only require 53x5 clock cycles, and 8 212-sample frames only require 106x8 clock cycles. Besides, the RDFT unit needs 14 reset clock cycle to initialize each frame computation. In total, one channel DTMF detection process would only require 1,233 clock cycles per window. On the contrary, based on the second-order Goertzel structure, one channel DTMF detection would require 2,346 clock cycles for each window, which is almost twice the latency of the proposed framework.

The high channel density DTMF detector as depicted in Fig. 9 consists of the recursive DFT (RDFT) units, an input unit, and a control unit. The behaviors of the above units are described as follows:

RDFT Unit: The RDFT unit as depicted in Fig. 9 consists of one pre-processing element and one recursive processing element (PE). The pre-processing element is able to provide the intermediary data s_k and r_k to the following recursive PE. Recalling (49), (55), (62), and (66), our proposed VLSI algorithm only needs $N/2$ clock cycles to accomplish each output data sequence.

Input Unit: The input unit is composed of a dual port SRAM that can store 318 complex data sequences. It could serve two sizes of input data buffer: 106 and 212 samples. According to the proper scheduling, the input unit can provide the dual data $x'[n]$ and $x'[N-1-n]$ for the pre-processing element of the RDFT unit.

Control Unit: The control unit not only plays the role of the data sequence controller but also a parameter controller, which feeds the proper coefficients to the RDFT units. In this paper, since the input data and output data of the proposed architecture are all controlled in the serial manner, the desired output data can be obtained for each $N/2$ clock cycles.

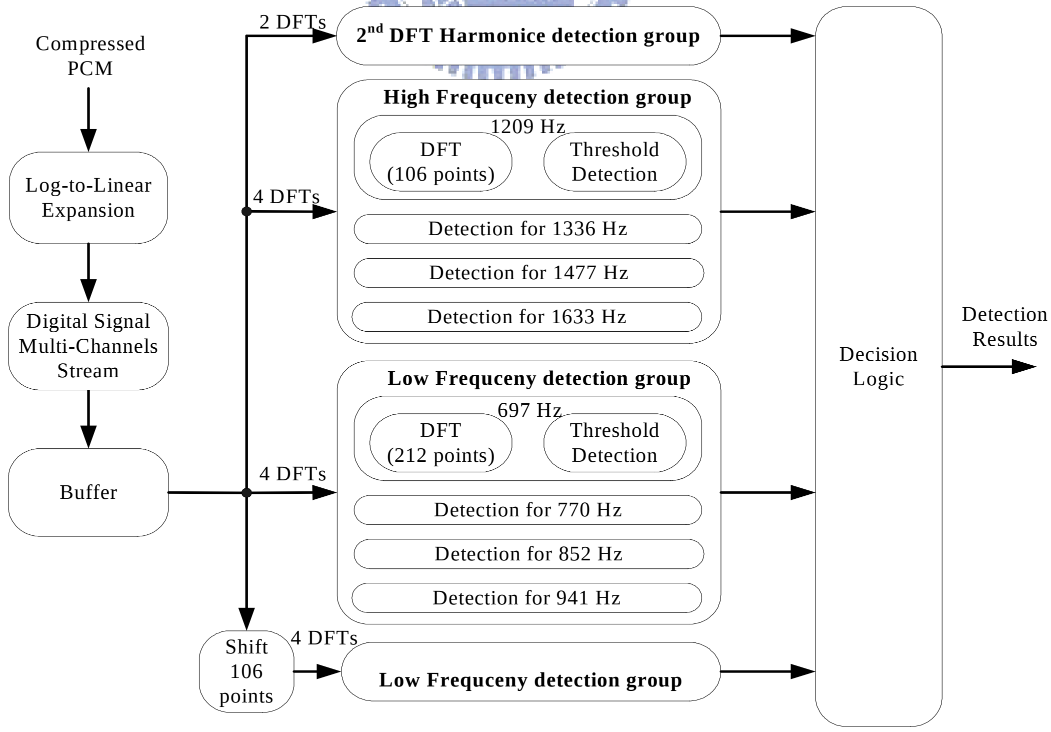


Fig. 8: Dataflow of the DTMF detection [21].

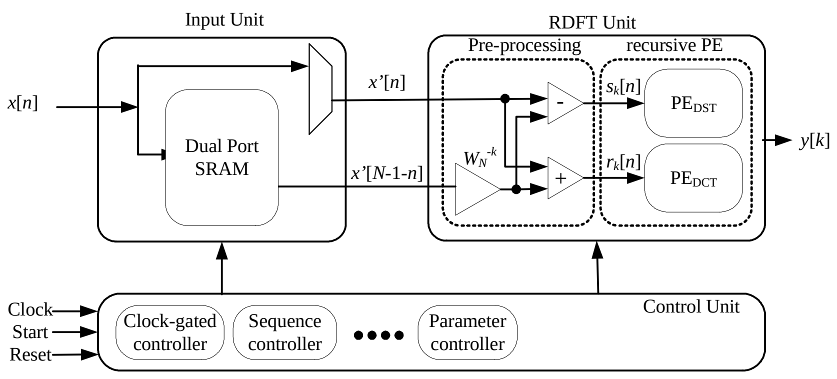


Fig. 9: Block diagram of the proposed high channel density DTMF architecture.

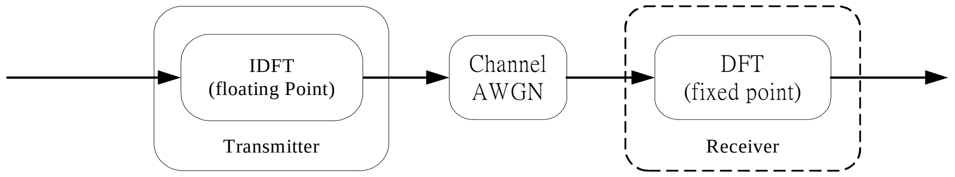


Fig. 10: Bit level SNR simulation environment.

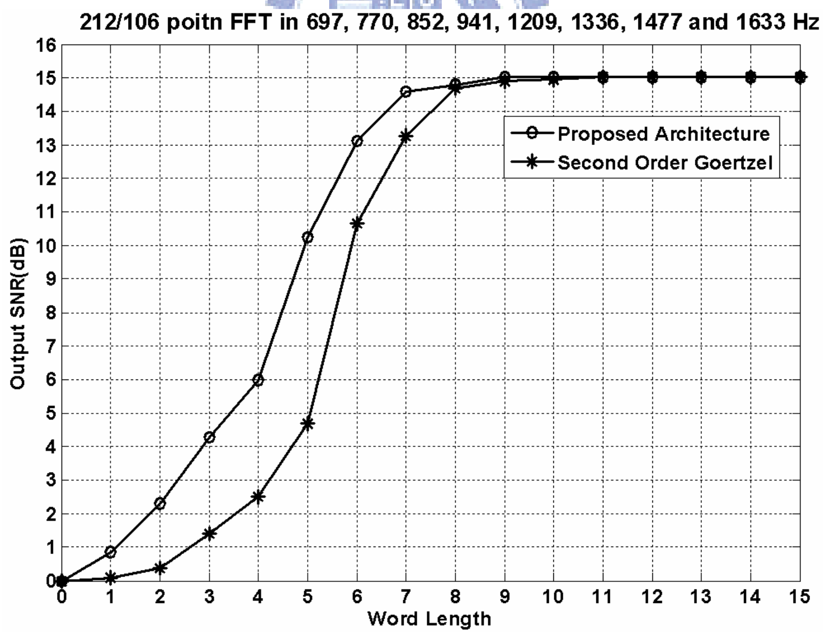


Fig. 11: Bit level SNR simulation results.

Next, we adopt the bit-level SNR simulation to estimate the appropriate word-length under the ITU specification [11] to further reduce the chip area and power consumption. We know that the DTMF detector must operate properly under 15dB SNR or higher. Thus, we set the simulation environment as depicted in Fig. 10 under 15dB with additive white Gaussian noise (AWGN) channel model. Then, we will only consider the DFT part in the receiver side for the DTMF detector. In Fig. 10, the input signal $x[n]$ passes thought IDFT block and then propagates through the channel, where the above operations run at floating point simulation. In the receiver side, the receiver signal is quantized into the fixed bits and performs the fixed-point DFT calculation. We perform the system simulation of 212/106-sample frames at the 8 DTMF signal frequency bins: 697, 770, 852, 941, 1209, 1336, 1477 and 1633 Hz as shown in Fig. 8. In Fig. 11, the x-axis and y-axis denote the data word-length and the whole system output SNR, respectively. We can observe that the output SNR will saturate as data word-length increases. It is manifest that the proposed recursive architecture only needs 9-bit resolution, which is less than 10-bit of the second-order Goertzel structure. That means we need less hardware resources to achieve the ITU performance requirements under our proposed architecture. In other words, if we select the same word-length for the proposed and Goertzel based designs, the former is able to offer the higher design margin for better system performance. In this case, because 3-bit design margin is sufficient, we choose the data word-length as 12-bit wide.

Concerning the chip implementation, our target is 212/106-point DFT/IDFT for high channel density DTMF detector [17-19]. As we know, the ITU timing specification indicates that the durations of DTMF signal detection and non-detection must be at least 40 ms and less than 23 ms, respectively. At a sampling rate of 8K Hz, a 106-sample frame size corresponds to a 13.3 ms window. After each window, the detected signal is compared to the last and second-to-last values. If the result of the new window is the same as the last, but different from the second-to-last, then a new valid DTMF signal has been found [13]. Recall that the proposed architecture requires 1233 clock cycles to finish one channel DTMF detection for each window. In this paper, the operating frequency and guard time are targeted at 20 MHz and 31.6 ms, respectively. That means we only need 61.65 μs (i.e., 1233x50 ns) to finish one window computation for one channel DTMF detection. Accompany with the DTMF FSM controller [13], the proposed design can detect up to 128-channel DTMF signals, which is superior to [12-14]. The implementation processes are as follows. First, the Cadence NC-Simulator is used as the Verilog functional verification, so the outputs from the RTL model are validated against a standard LabVIEW model. Then, the 212/106-point recursive DFT/IDFT architecture in which the internal word-length is 12-bit has been synthesized with the Design Compiler in TSMC 0.13 μm CMOS technology. After the post simulation, at the present stage, the critical path is 43.12 ns in TSMC 0.13μm CMOS process. Consequently, the proposed design is very suitable for DTMF detector system. The floorplan as well as the post-layout have been carried out using Astro. After the back-annotation from Start-RC extractor, the post-simulation has been issued by NC-Simulator to verify the functionality. The static timing check can be signed-off by PrimeTime. Finally, the power analysis and LVS can be done by Astro Rail and Dracula, respectively. For post layout, the core area is 0.18 mm^2. The chip characteristics listed in Table 2 shows that the average power dissipation of the proposed high channel density DTMF detector is 1.25 mW@20 MHz at 1.2V supply voltage. It is worth to notice that the proposed design could handle the 128 DTMF channel, that means each channel only consumes 9.77 µW after the division of 128. The microphotograph of the 212/106-point recursive DFT/IDFT core design as shown in Fig. 12 has been implemented as one hard IP (Intellectual Property). In this way, the proposed architecture and chip can be reused in the system-on-a-chip (SOC) platform. The proposed 212/106-point recursive DFT/IDFT design not only meets 40 ms timing specification for ITU standard, but also achieves the low power consumption due to the register-splitting scheme and smaller bit-width requirement compared with the design of [12-14].

Table 2: Chip Characteristics of the Proposed DTMF detector.

Maximum Channel	128
DFT Length (N)	212/106 points
Input Word Length (*w*)	12 bits
Critical Delay Time	43.12 ns
Chip Area	387 µm x 469 µm
Power Consumption per	9.77 µW@20
Process Technology	TSMC 0.13 µm

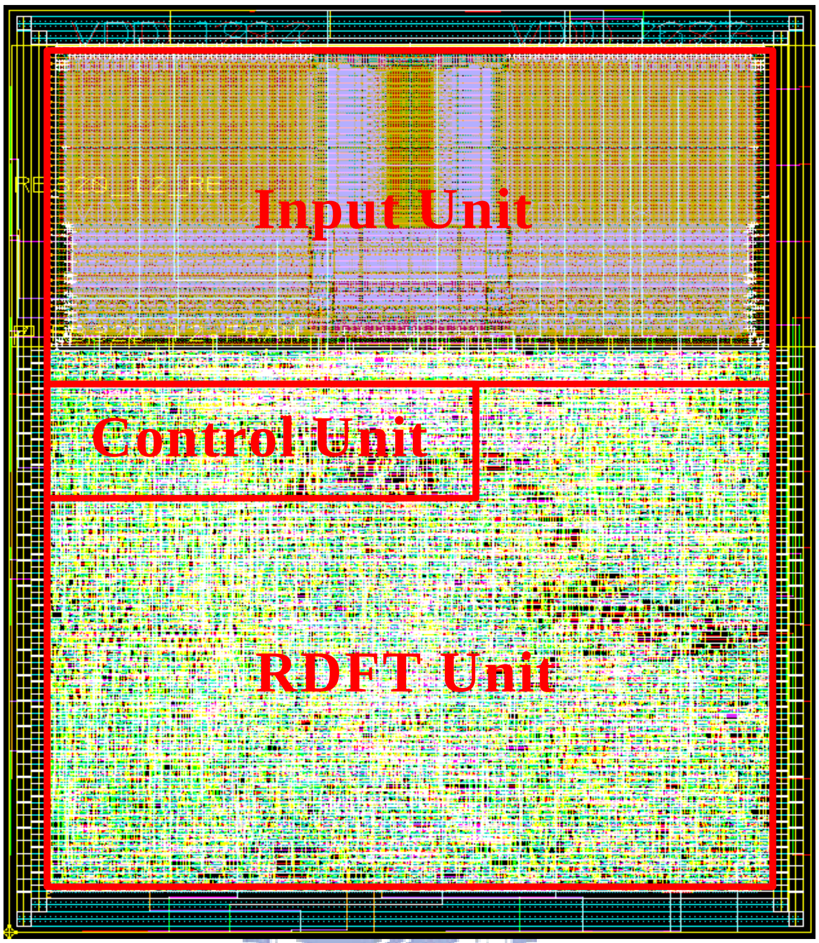


Fig. 12: The 212/106-point recursive DFT/IDFT chip layout.

3.3 The Comparison of Different Recursive DFT/IDFT Architecture

In this section, we give a comprehensive comparison result as listed in Table 3 in terms of the number of computation cycles for each DFT/IDFT output as well as *N*-point DFT/IDFT calculation, the maximum number of channel density, the clock period, and the number of real multipliers. Note that the operation time of a complex multiplication requires $T_m + T_a$. Our proposed work [66] based on the input strength reduction scheme can save half computation cycles for each DFT/IDFT output compared with the existing works [51, 52, 62] at the expense of slightly increased area cost. Note that we make a comparison between our proposed work and the best case design of [52], FAST fixed-coefficient recursive DFT (FFR-DFT), in terms of specific terminologies in Table 3. At the same time, the reference structure of [62] is the block diagram as shown in Fig. 9.2 of [62]. Compared with the results of the recursive algorithm in [53] which, for example, requires 2794 computational cycles to obtain all 64-point DFT outputs, the proposed core-type architecture requires 2048 computational cycles. In other words, our proposed work exploiting the input strength reduction scheme has the lowest computation cycles among existing structures [51-53, 62]. As a consequence, our proposed architecture is capable of providing the highest channel density in the DTMF communication system. From the implementation results, it is obviously seen that the channel amount of the proposed architecture is double compared with other designs [51, 52, 62]. Since exploiting the register-splitting scheme, the proposed one inherently has higher speed than the recursive structures of [51, 52, 62] and possesses the same operating frequency as that of our previous work [51]. According to the critical path comparison in Table 3, the proposed DFT/IDFT fabric owns $T_m + 2T_a$ clock period and the clock periods in [52, 53, 62] are of $T_m + 3T_a$, $T_m + 2T_a$, and $2T_m + 5T_a$, respectively. As mentioned in Section 3.1, the register-splitting scheme either achieves high speed or low power computation. In this article, we consider this technique for lowering the power consumption where the speed does not need to be increased [65]. In Table 3, if the architecture possesses a shorter clock period, less power consumption can be achieved while keeping the same clock rate. However, considering the hardware complexity, the proposed DFT/IDFT architecture requires two more multipliers than the previously proposed one [51]. Furthermore, based on the proposed work, we can easily construct a parallel-type recursive DFT/IDFT architecture for other applications such as the matching filter and equalizer. The parallel-type architecture can significantly reduce the number of computation cycles for N-point DFT/IDFT from $N^2/2$ to $\frac{N}{2}\cdot\left\lceil\frac{N}{P}\right\rceil$, where P is the number of RDFT and $\lceil\bullet\rceil$ indicates the minimum integer value greater than or equal to $\bullet$. Thus, the maximum throughput can be achieved. As a consequence, in Table 3, it reveals that our proposed architecture has characteristics of the lowest computation cycle (i.e., highest throughput), the maximum number of channel density, and power efficiency.

Table 3: Comparison Results among the Recursive DFT/IDFT Architectures.

Parameters	Second Order DFT/IDFT [2]	V-Ys' Structure [20] (Core Type)	Y-Cs' Structure [21] (FFR-DFT)	Proposed Work
# of Computation Cycles for Each $y[k]$ or $x[n]$	N	N	N	$N/2$
# of Computation Cycles for N-Point DFT/IDFT	N^2	N^2	N^2	$N^2/2$
Maximum of Channel Density (in TSMC 0.13 μm)	64	64	64	128
Clock Period	T_m+3T_a	T_m+2T_a	$2T_m+5T_a$	T_m+2T_a
# of Real Multipliers	6	4	6 (Pre-processing Excluded)	6 (Pre-processing Excluded)

3.4 Summary

One new recursive DFT/IDFT algorithm and architecture based on a hybrid of input strength reduction scheme, the Chebyshev polynomial and register-splitting scheme is devised in this framework. The analyzed results show that the proposed VLSI algorithm leads to the fewest computation cycle and the highest throughput rate. Moreover, the proposed 212/106-point recursive DFT/IDFT chip design has been successfully implemented in 0.13 μm CMOS technology and possesses the power-efficiency consumption of 9.77 uW@20 MHz at 1.2V supply voltage for each channel. These features guarantee that the proposed high-throughput and power-efficient VLSI architecture is certainly amenable to high channel density DTMF systems.

Chapter 4 Effective FFT/IFFT Processors for MIMO-OFDM WLAN Systems

In this chapter, we adopts the novel radix-2/8 algorithm, which is different from the conventional radix-2/8 algorithm [32, 59, 60], to further reduce the constant multiplier requirement in the proposed retrenched 8-point FFT (R8-FFT) unit. Accompany with the multiplier after write (MAW) scheme [57], this work adopts the novel radix-2/8 algorithm to devise two 64-points FFT/IFFT architectures, radix-2/8 multiple-path delay feedback (R28MDF) and radix-2/8 multiple-path delay commutator (R28MDC), for the high throughput rate system of 2R and 4R, respectively. A detailed comparison of the 64-point FFT/IFFT processors among several existing chips has been presented for the 2×2 MIMO-OFDM system, revealing that the new R28MDF implementation achieves the low complex multiplicative complexity, high butterfly utilization, low hardware cost and appropriate throughput rate. In the 4×4 MIMO-OFDM system, the proposed R28MDC implementation further applies the fully pipeline architecture to achieve 100% utilization of the complex multipliers, adders and memory. Comparison results indicate that the R28MDC architecture achieves the lower multiplicative complexity and lower chip cost than the R4MDC [40, 61] and other pipeline FFT/IFFT architectures. Thus, the proposed R28MDF and R28MDC architectures clearly achieve the high efficiency advantages for the 2×2 and 4×4 MIMO-OFDM WLAN application, respectively. The organization of this chapter is listed as follows. Section 3.1 describes the proposed modified radix -2/8 FFT/IFFT algorithm. Section 4.2 then discusses the corresponding R28MDF and R28MDC fabrics and the detailed timing considerations. The implementation issues are discussed in Section 4.3. Section 4.4 summarizes the comparison results of the 64-point FFT/IFFT architectures for the 2×2 and 4×4 MIMO-OFDM system. Conclusions are finally drawn in Section 4.5.

4.1 The Proposed Modified Radix-2/8 FFT/IFFT Algorithm

The discrete Fourier transform (DFT) of the N-point input $X[n]$ is given by

$$Z[k]=\sum_{n=0}^{N-1} X[n]\cdot W_N^{kn}, \tag{69}$$

where $W_N = e^{-j2\pi/N}$ and Z[k] represents the DFT output sequences for $0 \le k \le N-1$. Based on the decomposition of the radix-8 algorithm, (69) can be rewritten as

$$Z[s+Tt]=\sum_{l=0}^{M-1}\left[W_{MT}^{sl}\cdot\sum_{m=0}^{T-1} X\left(l+Mm\right)\cdot W_T^{sm}\right]\cdot W_M^{lt}\ . \tag{70}$$

Equation (4.2) indicates that the 64-point DFT can be separated into two-dimensional 8-point FFTs, where $M=T=8$. The 8-point FFT computation in (70) can then be written as:

$$\begin{bmatrix} Y_0\\ Y_1\\ Y_2\\ Y_3\\ Y_4\\ Y_5\\ Y_6\\ Y_7 \end{bmatrix}=\begin{bmatrix} W_8{}^0 & W_8{}^0 & W_8{}^0 & W_8{}^0 & W_8{}^0 & W_8{}^0 & W_8{}^0 & W_8{}^0\\ W_8{}^0 & W_8{}^1 & W_8{}^2 & W_8{}^3 & W_8{}^4 & W_8{}^5 & W_8{}^6 & W_8{}^7\\ W_8{}^0 & W_8{}^2 & W_8{}^4 & W_8{}^6 & W_8{}^0 & W_8{}^2 & W_8{}^4 & W_8{}^6\\ W_8{}^0 & W_8{}^3 & W_8{}^6 & W_8{}^1 & W_8{}^4 & W_8{}^7 & W_8{}^2 & W_8{}^5\\ W_8{}^0 & W_8{}^4 & W_8{}^0 & W_8{}^4 & W_8{}^0 & W_8{}^4 & W_8{}^0 & W_8{}^4\\ W_8{}^0 & W_8{}^5 & W_8{}^2 & W_8{}^7 & W_8{}^4 & W_8{}^1 & W_8{}^6 & W_8{}^3\\ W_8{}^0 & W_8{}^6 & W_8{}^4 & W_8{}^2 & W_8{}^0 & W_8{}^6 & W_8{}^4 & W_8{}^2\\ W_8{}^0 & W_8{}^7 & W_8{}^6 & W_8{}^5 & W_8{}^4 & W_8{}^3 & W_8{}^2 & W_8{}^1 \end{bmatrix}\cdot\begin{bmatrix} X_0\\ X_1\\ X_2\\ X_3\\ X_4\\ X_5\\ X_6\\ X_7 \end{bmatrix} \tag{71}$$

The notations X_i and Y_i indicate the input data and output result of the 8-point FFT computation, respectively, where $0 \le i \le 7$. After eliminating the 180° and 90° redundancies of twiddle factors, another simple matrix result can easily be calculated. By re-ordering the output sequence, the alternative matrix representation (72) can be obtained.

$$\begin{bmatrix} Y_0\\ Y_1\\ Y_4\\ Y_5\\ Y_2\\ Y_3\\ Y_6\\ Y_7 \end{bmatrix}=\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0 & W_8^1 & 0\\ 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0 & -W_8^1 & 0\\ 0 & 1 & 0 & 0 & 0 & -j & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -jW_8^1\\ 0 & 1 & 0 & 0 & 0 & j & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & jW_8^1 \end{bmatrix}\cdot\begin{bmatrix} (X_0+X_4)+(X_2+X_6)\\ (X_0+X_4)-(X_2+X_6)\\ (X_0-X_4)-j(X_2-X_6)\\ (X_0-X_4)+j(X_2-X_6)\\ (X_1+X_5)+(X_3+X_7)\\ (X_1+X_5)-(X_3+X_7)\\ (X_1-X_5)-j(X_3-X_7)\\ (X_1-X_5)+j(X_3-X_7) \end{bmatrix}$$

(72)

Clearly, the four quarters of the transform matrix (72) all exhibit the same symmetric property. Thus, the 8-point FFT transform matrix in (72) can be decomposed as

$$\begin{bmatrix} Y_0 \\ Y_1 \end{bmatrix} = G_{1(FFT)} + H_{1(FFT)}, \quad \begin{bmatrix} Y_4 \\ Y_5 \end{bmatrix} = G_{1(FFT)} - H_{1(FFT)} \tag{73}$$

$$\begin{bmatrix} Y_2 \\ Y_3 \end{bmatrix} = G_{2(FFT)} + H_{2(FFT)}, \quad \begin{bmatrix} Y_6 \\ Y_7 \end{bmatrix} = G_{2(FFT)} - H_{2(FFT)} \tag{74}$$

where

$$G_{1(FFT)} = \begin{bmatrix} (X_0 + X_4) + (X_2 + X_6) \\ (X_0 - X_4) - j(X_2 - X_6) \end{bmatrix} \tag{75a}$$

$$G_{2(FFT)} = \begin{bmatrix} (X_0 + X_4) - (X_2 + X_6) \\ (X_0 - X_4) + j(X_2 - X_6) \end{bmatrix} \tag{75b}$$

$$H_{1(FFT)} = \begin{bmatrix} 1 & W_8^1 \end{bmatrix} \cdot \begin{bmatrix} (X_1 + X_5) + (X_3 + X_7) \\ (X_1 - X_5) - j(X_3 - X_7) \end{bmatrix} \tag{75c}$$

$$H_{2(FFT)} = -j \cdot \begin{bmatrix} 1 & W_8^1 \end{bmatrix} \cdot \begin{bmatrix} (X_1 + X_5) - (X_3 + X_7) \\ (X_1 - X_5) + j(X_3 - X_7) \end{bmatrix} = \begin{bmatrix} 1 & W_8^1 \end{bmatrix} \cdot \begin{bmatrix} -j(X_1 + X_5) + j(X_3 + X_7) \\ -j(X_1 - X_5) + (X_3 - X_7) \end{bmatrix} \tag{75d}$$

In the similar behaviors, the IFFT equation representation can easily be calculated. The detailed expression is given in [57]. The difference between the FFT and IFFT equations is the sign bit in the matrixes G_1, G_2, H_1 and H_2, which can easily implement the FFT and IFFT processor in a single chip. Significantly, the presented algorithm has the similar results as the conventional radix-2/8 algorithm with the parameter "*q*=1, *m*=6" [59]. The conventional radix-2/8 algorithm [32, 59, 60] splits the 8-point FFT computation results into odd-half and even-half components in (72). Then, the butterfly computation requires two constant multiplications, which can be calculated by an "L" shaped butterfly. To implement radix-8 FFT algorithm more efficiently, the purposed 8-point butterfly computations decomposed into another index map, which is different from the conventional decimation-in-frequency (DIF) based radix-2/8 algorithm. The derivation results of modified radix-2/8 algorithm in (72) indicates that W_8^3 can be replaced by $-jW_8^1$. Thus, the "L" shaped butterfly can be modified as illustrated in Fig. 13.

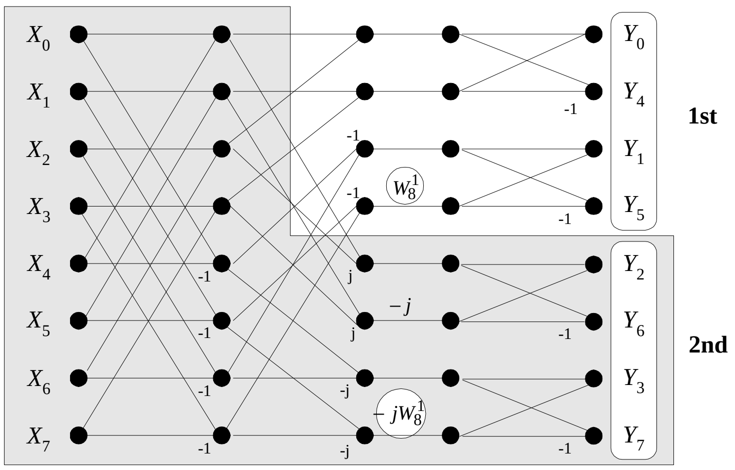


Fig. 13: The "L" shaped butterfly of novel radix-2/8 FFT algorithm.

4.2 The Proposed MIMO-FFT Architecture

Applying the proposed radix-2/8 algorithm, we propose two 64-point R28MDF and R28MDC FFT/IFFT architecture, for the high throughput rate system of 2R and 4R, respectively.

4.2.1 R28MDF-based 64-Point FFT/IFFT Processor for 2×2 MIMO-OFDM system

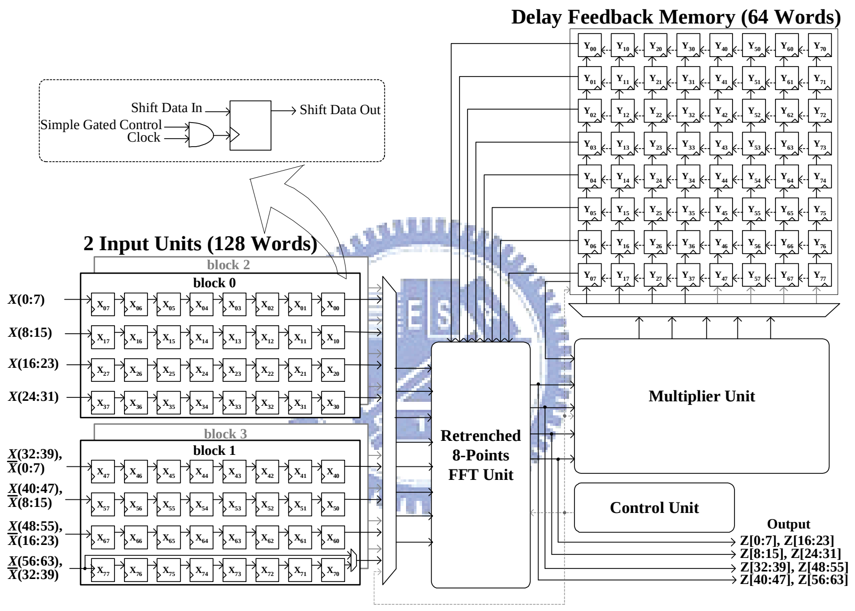


Fig. 14: Block diagram of the proposed R28MDF-based 64-point FFT/IFFT architecture for 2X2 MIMO-OFDM system.

The R28MDF design comprises two input units (IU), one retrenched 8-point FFT (R8-FFT) unit, one multiplier unit (MU), one delay feedback memory (DFM) and one control unit (CU) as shown in Fig. 14. The detailed operations of each building unit are described as follows.

IU: The IU contains one register bank, which can store 64 complex 16-bit word-length data. These 64 complex registers are split into 8 parallel shift-register lines as illustrated in Fig. 14. Each shift-register line can be easily controlled independently by the simple clock-gated controller. In Figs. 16 and 18, the subscripts of each element are represented as radix-8 based notation. Figure 14 shows that the proposed R28MDF-based serial blockwise architecture contains two input units to store two channel input data for the 2×2

MIMO-OFDM system, represented as X and $\overline{X}$, to realize the functionality of the input buffer as discussed in the chapter 1. To prevent the input data overflow, the R28MDF architecture groups these 16 shift-register lines into four blocks, namely, block0, block1, block2 and block3, each of which contains four parallel shift-register lines. In the consequent timing frames, the proposed input unit applies two different combinations of these four blocks to store two-channel input data to prevent input data overflow as depicted in Fig. 15. Each timing frame contains 64 clock cycles. In Fig. 15, X(0:63), X(64:127) and X(128:191) denote input data in the first, second and third timing frames, respectively.

In the first timing frame, block0 and block1 are utilized to store input data X(0:31) and X(32:63), block2 and block3 are used to store input data $\overline{X}(0:31)$ and $\overline{X}(32:63)$ as shown in Fig. 15. The proposed R28MDF architecture requires 32 cycles to complete the 64-point FFT/IFFT computations, which is described in detail in the following subsection. In the preceding 32 cycles of the second timing frame, the data X(0:63) in block0 and block1 are pushed into the R8-FFT unit in parallel. Simultaneously, input data X(64:95) and $\overline{X}(64:95)$ can seamlessly replace the data contexts in block0 and block1. During cycles 96–127, the proposed design completes the 64-point FFT/IFFT computation of data $\overline{X}(0:63)$ in block2 and block3, and input data X(96:127) and $\overline{X}(96:127)$ concurrently replace the data contexts in block2 and block3. Based on these block-based input unit architectures with appropriate multiplexing control, two channel input data can be easily pushed to the R8-FFT unit using128 words shift-registers as depicted in Fig. 14.

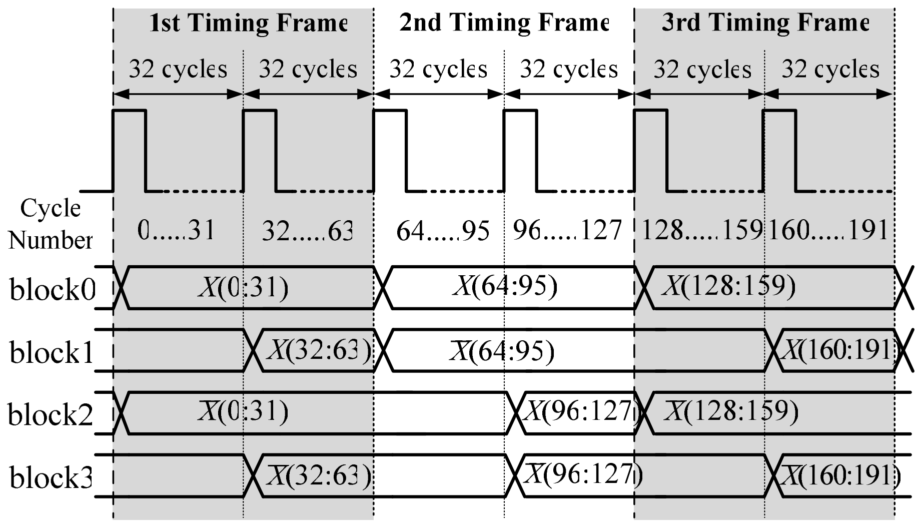


Fig. 15: The timing sequence of the purposed block based input unit.

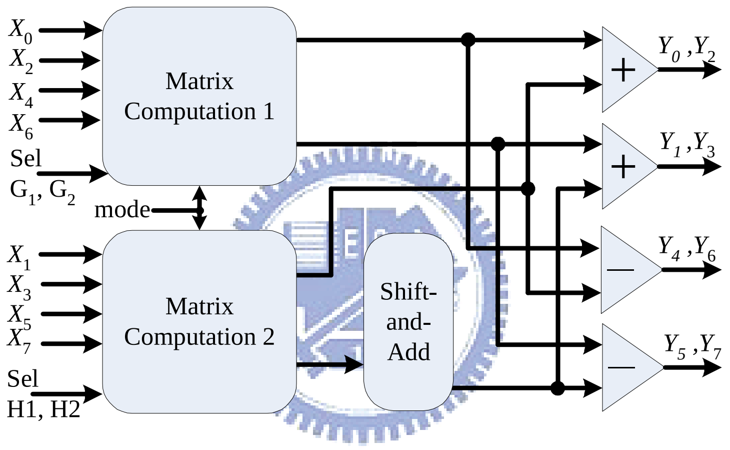


Fig. 16: Block diagram of the proposed R8-FFT/IFFT unit.

R8-FFT unit: By sharing one constant multiplier in the radix-8 based butterfly in two clock cycles, the proposed equation (72) could produce a low cost and high-efficiency 8-point FFT/IFFT butterfly kernel as illustrated in Fig. 16, called the R8-FFT unit. The constant multiplier in the R8-FFT unit is fully implemented with the shift-and-add circuits, while the proposed parallel type multiplier unit (MU) is fully implemented with eight constant multipliers. The IFFT architecture can be easily obtained by controlling the mode signal in Fig. 16, and the operations of IFFT are similar to those of FFT. The detailed description is omitted here.

MU: The MU as illustrated in Fig. 17 comprises eight constant multipliers to realize different multiplications of the W_{MT}^{sl} in (70). For the purpose of completing the 64-point FFT/IFFT computation in (70), the 64-point FFT/IFFT operation sequence can be separated into two operational stages, namely the multiplication stage (MS) and the output stage (OS), as illustrated in Figs. 18(a) and 18(b), where the number inside brackets denotes the usage of the constant name in the MU. Notably, the MU can only be adopted in the MS. Thus, the input ports of the MU should be gated during the OS to further reduce the power consumption. The MU contains five independent multiplication pair-ports in parallel, which has one more port than the modified R8MDC design [41]. The modified R8MDC design has to been halted for five clock cycles during FFT computation because of the resource conflictions. The proposed architecture adopts this port to resolve the performance degradation. The conflict clearly occurs in four different clock cycles, with clock cycle numbers of 6, 10, 11 and 14, as revealed in Fig. 18(a). In the clock cycles 8, 11, 12, 13 and 16, the fifth pair-port P(4), could re-serve the multiplication to re-fill the data $R_{62}(4)$, $R_{64}(8)$, $R_{45}(4)$, $R_{74}(4)$ and $R_{66}(4)$ to DFM, which is called MAW. Using the MAW method, the proposed architecture is capable of completing the computation in 16 clock cycles for each operational stage. The R28MDF architecture only needs 32 clock cycles to complete the two operational stages. The R28MDF architecture can thus complete two 64-Point FFT/IFFT computations in 64 clock cycles. Hence, the proposed R28MDF architecture can achieve a higher throughput rate of 2R, which is the twice that of the R2^{2}SDF architecture as illustrated in Fig. 18(c).

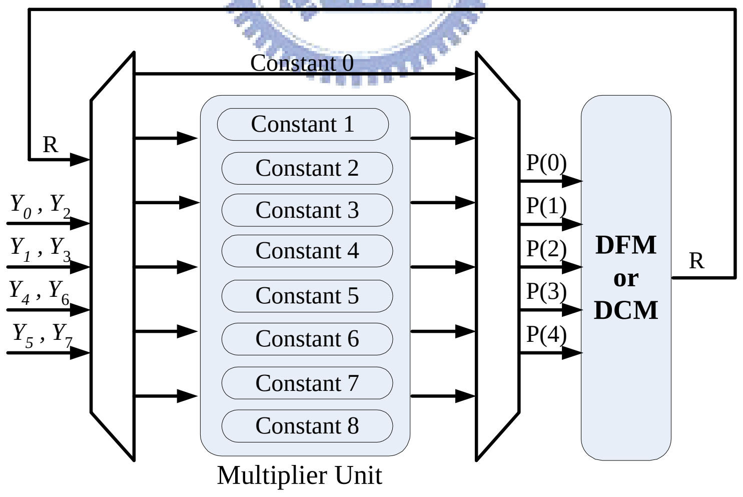
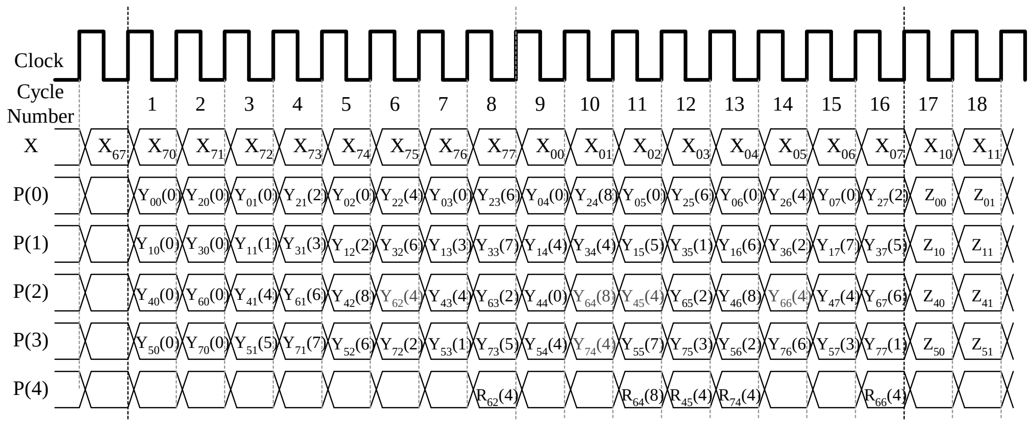
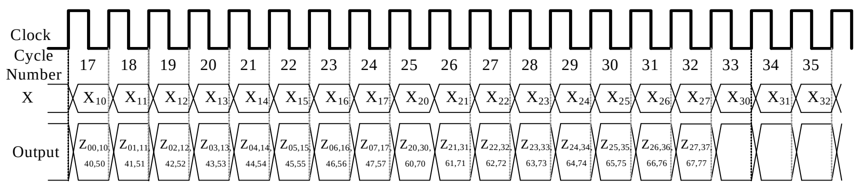


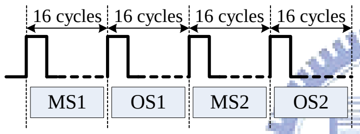
Fig. 17: Block diagram of the proposed MAW-based multiplier unit.

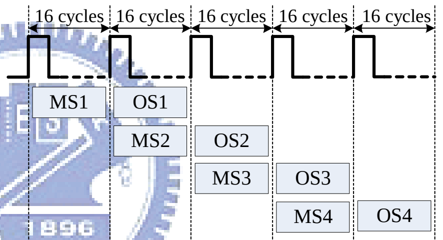


(a) The first stage: Multiplication Stage/



(b) The second stage: Output Stage.





(c) The timing sequence of R28MDF design.

(d) The pipeline timing sequence of R28MDC design.

Fig. 18: The timing sequence of the proposedR28MDF and R28MDC architectures.

DFM: The DFM contains one register bank, which can store 64 complex 16-bit wordlength data. The DFM is adopted to store the intermediate coefficient parameters from R8-FFT unit, as illustrated in Fig. 14. To save power, the DFM is built by one matrix based buffer architecture with the proper-gated control, as illustrated in Fig. 14.

CU: The CU contains a 6-bit master counter to manage the entire procedures, and gates the unused parts during the redundant period to minize power consumption. Although the proposed CU should pay very small area effort to realize the MAW, the proposed design still raises the throughput rate of 2R with only one constant multiplier.

4.2.2 R28MDC-based 64-Point Pipeline FFT/IFFT Processor for 4×4 MIMO-OFDM System

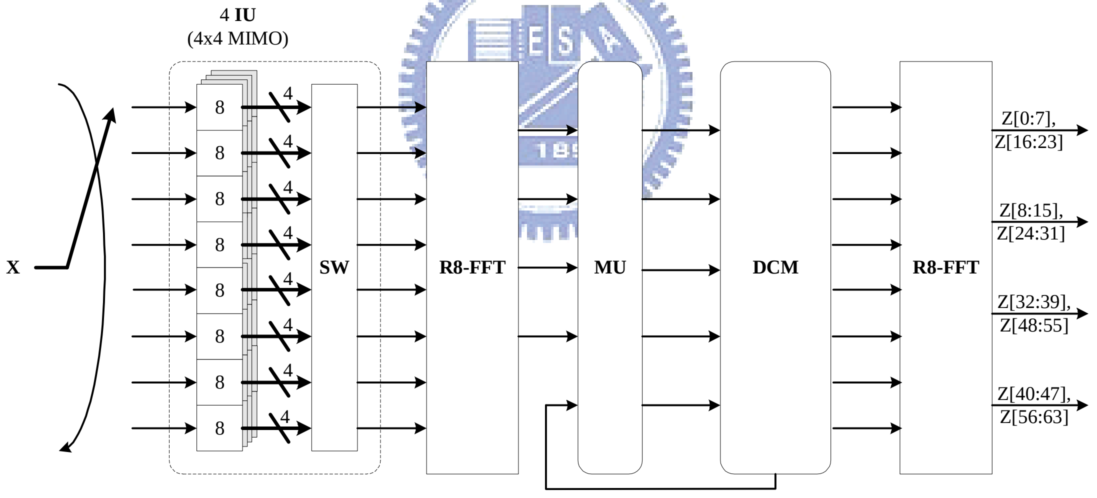


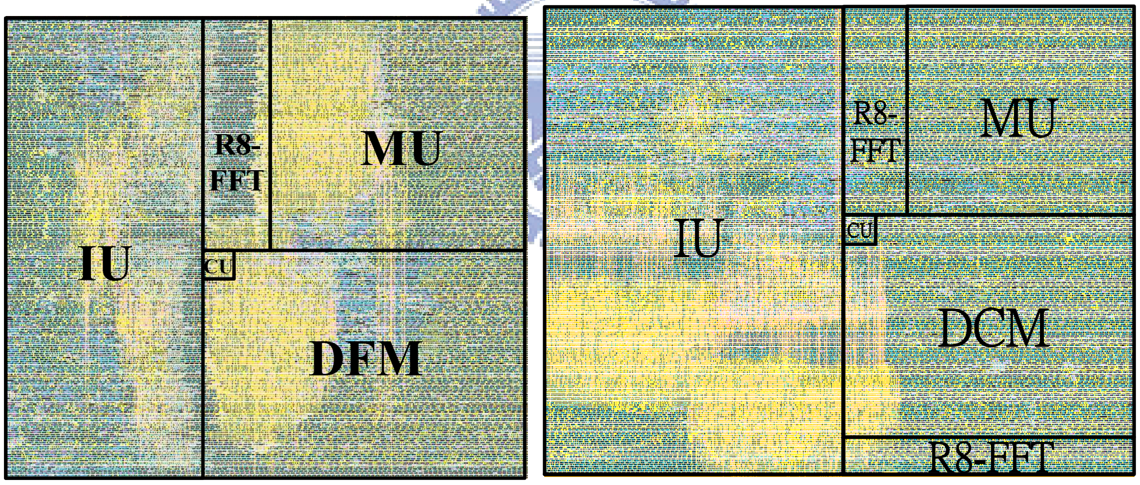
Fig. 19: Block diagram of the proposed R28MDC-based 64-point FFT/IFFT architecture for 4X4 MIMO-OFDM system.

For the 4×4 MIMO-OFDM WLAN application, another R28MDC design based on pipeline architecture is presented to further raise the throughput rate to 4R, which is double than that of the R28MDF architecture. The R28MDC design comprises four input units (IU), two retrenched 8-point FFT (R8-FFT) units, one multiplier unit (MU), one delay commutator memory (DCM) and one control unit (CU) as shown in Fig. 19. Additionally, the R28MDC architecture has four IUs, allowing it to store four-channel data for the 4×4 MIMO-OFDM system. Based on the block-based input buffer architecture, which is similar to the R28MDF design, the R28MDC architecture groups 32 shift-register lines into 16 blocks. Each block contains two parallel shift-register lines. Applying two different combinations of these 16 blocks with a simple clock gated controller in CU, these four IUs can prevent input data overflow in the consequent timing frames for the 4×4 MIMO-OFDM system. For the 2×2 MIMO-OFDM WLAN system, the R28MDF design apply the feedback path to reduce the number of R8-FFT unit to only one with the 100% butterfly utilization rate as illustrated in Fig. 14. Compared with the R28MDF architecture, the R28MDC architecture adopts the feedforward path rather than the feedback path as illustrated in Fig. 19. Thus, the feedback-type memory architecture of the DFM is replaced by the feedforward-type delay commutator memory (DCM) architecture. Additionally, another R8-FFT unit should be inserted following the DCM. Otherwise, the structures of the R8-FFT unit and the MU are the same as those in the R28MDF architecture, but two intermediate multiplexes have been eliminated.

The MAW scheme can finish the computation period of each stage in the R28MDC architecture within 16 clock cycles without any performance degradation. In the R28MDF architecture, the same R8-FFT unit should execute the computation of MS and OS with the feedback path. The throughput rate of the R28MDF architecture is 2R. However, the first R8-FFT unit only performs the MS computation, and the second R8-FFT unit only performs the OS computation in the R28MDC architecture. Thus the R28MDC architecture can provide the double throughput rate of the R28MDF. Hence, the four channels computation can be completed in 64 clock cycles for the 4×4 MIMO-OFDM system, as illustrated in Fig. 18(d). The proposed R28MDC architecture attains a high throughput rate of 4R, which is the same as that of the R4MDC design.

4.3 Circuit Implementation

This work presents the R28MDF and R28MDC implementations for 2×2 and 4×4 MIMO-OFDM WLAN applications [23], respectively. As is well known, the processing time of 64-point FFT/IFFT for *IEEE* 802.11n standard has to be within 3.2μs without the guard interval [23]. The proposed 64-point FFT/IFFT design can maintain an appropriate throughput in the sampling data frequency of 20MHz for the MIMO-OFDM system. The R28MDF and R28MDC design thus achieves throughput rates of 2R and 4R to meet the *IEEE* 802.11n standard, respectively. Following functional verification by MATLAB, the proposed design was modeled in Verilog and verified using an NC-Verilog simulator. In this investigation, the proposed design with an internal word length of 16 bits was synthesized using a Design Complier based on TSMC 0.13μm 1P8M CMOS technology. The floorplan and the post-layout were performed by Astro. Following the back-annotation from Start-RC extractor, the post-simulation was performed by the NC-Verilog simulator to verify the functionality. The static timing check was signed-off by PrimeTime. Finally, the power analysis was performed by Astro Rail. Figure 20(a) and. 20(b) show the core layouts of the R28MDF and R28MDC designs, respectively. For the post layout, the core area of R28MDF was 0.75 mm^2, which includes power rings and power straps as depicted in Fig. 20(a). The average power dissipation of the proposed R28MDF design was 19.42mW@20 MHz at 1.2V supply voltage. The core area of R28MDC was 0.98 mm^2, as depicted in Fig. 20(b), and the power dissipation was 23.57mW@20 MHz at 1.2V supply voltage. Table 4 lists the gate count usage of each building unit. In Table 4, the small gate count usages in CU show that the small area expense for supporting the MAW can be ignored. Significantly, the matrix-based DFM architecture in the R28MDF design reduces routing complexity compared with the serial architecture in [57]. The routing area of the physical design in our previous scheme was reduced. Base on implementation results, the R28MDC implementation further has a smaller routing area than R28MDF implementation using the feedforward path architecture. Following the back-annotation, the static timing analyses indicate that the critical paths of the R28MDF and R28MDC design are 48.3ns and 47.8ns, respectively. The implementation results demonstrate that the proposed 64-point FFT/IFFT design satisfies the 3.2μs timing specification of *IEEE* 802.11n standard for the 2×2 and 4×4 MIMO-OFDM wireless applications.



(a) The R28MDF implementation. (b) The R28MDC implementation.

Fig. 20: Layout view of the proposed 64-point FFT/IFFT processors.

Table 4: Area usage of each building block in the proposed R28MDF and R28MDC design.

Implementation	**IU**	**R8-FFT**	**MU**	**DFM/DCM**	**CU**
R28MDF	37.6 %	6.6 %	27 %	27.9 %	0.9 %
R28MDC	53.2 %	9.2 %	17.2 %	20 %	0.4 %

4.4 The Comparison Discussion of MIMO-FFT Architecture

Considering the most efficient pipeline FFT processor in a single-input single-output OFDM (SISO-OFDM) WLAN application, He *et al.* have presented several reliable architectures and the detailed comparison of their hardware costs [31]. The comparison of these architectures indicates that the radix-2^2 single-path delay feedback (R2^2SDF) has the highest 50% butterfly utilization and lowest hardware resource consumption [31, 34]. However, the radix-2^2 based algorithm has a higher complex multiplicative complexity than high-radix and other mixed-radix FFT algorithms, as revealed in Table 1. Furthermore, the SDF based architecture has the lowest throughput rate of R, which can not meet the requirements of the MIMO-OFDM applications. Considering the most efficient pipeline FFT processor in the MIMO-OFDM WLAN applications, the comparison results of [26] indicates that the R4MDC architecture meets the most efficient 64-points FFT/IFFT processor for the 4×4 MIMO-OFDM WLAN system. Although several R4MDC based 64-point FFT chips have been discussed [40, 61, 72], only the design of Swartzlander *et al.* [40] can operate at the data sampling frequency in the 4×4 MIMO-OFDM systems. Notably, Hui *et al.* [56] proposed a digit-serial architecture base on radix-4 decomposition, with higher hardware utilization (100%) than the R4MDC based design [40] in the SISO-OFDM system. Hui *et al.* made good tradeoffs between the digit size and throughput rate in the SISO system. However, the radix-4 based design has a higher complex multiplicative complexity than high-radix and other mixed-radix FFT algorithms, too. This work focuses on the high throughput rate design with the low multiplicative complexity to fit the requirements of 2×2 and 4×4 MIMO-OFDM systems.

This section presents detailed comparisons among the two proposed architectures, R28MDF and R28MDC, and several famous FFT architectures in the 2×2 and 4×4 MIMO-OFDM systems. An effective design is well to be dictated by considerations on area, timing, power consumption and easily reuse. In this investigation, the systems were compared using five indices —MIMO-FFT architecture, complex multiplicative complexity, throughput rate, utilization and cost— to assess the effectiveness of FFT/IFFT processors. For the purpose of estimating the area index between the different architectures, the conventional comparative methodology [26] with the unit of equivalent adders was adopted. Based on the implementation results of our process, one complex multiplier is equivalent to 50 complex adders if it utilizes 16-bit precision and the scheme of three real multiplications and five real additions. The 16-bit complex memory was converted to 1.3 complex adders. The area report of the logic synthesis tool demonstrates that one proposed MU is considered to equal 3.2 complex multipliers. Furthermore, the area of proposed constant multiplier is equivalent to one-eighth times that of the proposed MU. Restated, one constant multiplier is approximately equivalent to 0.4 complex multipliers.

4.4.1 2×2 MIMO-OFDM WLAN Application

Table 5: Comparison results of the 64-point FFT/IFFT chip designs in 2x2 MIMO-OFDM system.

Architecture	MIMO-FFT architecture (Frequency, MHz)	Complex Multipli-cation #	Through-put rate	Butterfly Utilization	Cost			
					ROM #	complex multipliers #	constant multipliers #	Area without memory (Area with memory)
Modified R2^{2}SDF [34]	Parallel Multi-Path (20)	76	R	50%	2	4	0	224 (390.4)
R28SDF [32]	Parallel Multi-Path (20)	48	R	25%	4	4	8	304 (470.4)
R2MDC [67]	Serial Blockwise (20)	98	2R	100 %	4	4	0	424 (834.8)
R4MDC [40]	Serial Blockwise (20)	76	4R	50%	6	6	0	324 (776.4)
Modified R4MDC [61]	Serial Multi-Stream (80)	76	4R	50%	4	4	0	340 (1120)
Modified R8MDC [41]	Serial Blockwise (20)	48	5.33R	25%	0	3.2	4	228 (709)
Proposed R28MDF	Serial Blockwise (20)	48	2R	100 %	0	3.2	1	197 (446.6)

Table 5 presents the comprehensive comparison results of seven existing 64-point FFT/IFFT processors and the proposed R28MDF design in terms of MIMO-FFT architecture, complex multiplicative complexity, throughput rate, butterfly utilization, the number of ROM/complex multipliers/constant multipliers and the area index. Table 5 shows that the proposed R28MDF and R8MDC design achieve the lowest complex multiplicative complexity among the tested design. In terms of butterfly utilization, the proposed R28MDF design achieved the highest butterfly utilization (100%) among those tested. The R28SDF [32] and R2^{2}SDF [34] designs clearly have lowest throughput rates of R than other designs. Significantly, the R8MDC-based FFT/IFFT architecture in [41] has two butterfly stages, which only needs 12 and 11 clock cycles respectively. Base on the serial blockwise architecture, the parallel input data for each butterfly stages in [41] could be provided simultaneously to achieve the higher throughput rate. Table 5 shows that the modified R8MDC [41] and R4MDC [40] design could attain higher throughput rates of 5.33R and 4R, respectively, but both of them have lower butterfly utilization and higher chip cost than the proposed R28MDF design. Sansaloni *et al.* [26] indicated that the MIMO-FFT processor with throughput rates of 2R and 4R with the least amount of hardware was more appropriate than other architectures for 2×2 and 4×4 MIMO-OFDM applications, respectively.

Based on the serial blockwise architecture, the proposed R28MDF design should incur a small cost penalty on two IUs and one DFM memory in the 2×2 MIMO-OFDM system. When considering the memory area, the cost of the R28MDF design increases the area index to 14.4% higher than that obtained with the R2^{2}SDF [34] design. However, the R^{2}SDF design increases the multiplicative complexity by 58.3% and reduces the butterfly utilization to 50% of that of the R28MDF design. Furthermore, the proposed design and that of Maharatna *et al.* [41], which only adopt one parallel type multiplier unit, do not require any coefficient ROM. Following comprehensive comparison between different architectures, this investigation demonstrates that the proposed R28MDF implementation minimizes the chip cost problem associated with the R8MDC, R4MDC architectures, low throughput rate problem of R2^{2}SDF and R28SDF architectures, and the high multiplicative complexity problem of R2^{2}SDF and R2MDC architectures. Thus, the proposed R28MDF design makes an effective tradeoff between complex multiplicative complexity, throughput rate, butterfly utilization and cost for the 2×2 MIMO-OFDM application.

4.4.2 4×4 MIMO-OFDM WLAN Application

For a 4×4 MIMO-OFDM system, Table 6 presents the comprehensive comparison result of several pipeline FFT/IFFT architectures in terms of the MIMO-FFT architecture, throughput rate, complex multiplicative complexity, the utilization of all components, the number of complex multipliers/complex adders/memory size and the area index of the entire system. Table 6 shows that the proposed R28MDC design achieves the lowest complex multiplicative complexity among the tested design. Furthermore, the proposed R28MDC and R4MDC [40] achieved the highest utilization (100%) for all components; thus R28MDC and R4MDC design were the best among all pipeline architectures tested for the 4×4 MIMO-OFDM application. Although the R4MDC architecture [40] achieved 100% utilization for all components, it also resulted in a chip area 25.6% larger than that of the R28MDC architecture, when considering the memory cost. Regardless of whether memory cost is considered, the proposed R28MDC architecture had the smallest chip area among all pipeline architectures tested in the 4×4 MIMO-OFDM system. The R28MDC architecture did not require any coefficient ROM, also representing an improvement over the R4MDC architecture. Then, the R28MDC architecture achieved the lowest complex multiplicative complexity, appropriate throughput of 4R, highest utilization for all components and lowest chip cost, making it very suitable for the 4×4 WLAN MIMO-OFDM application.

Table 6: Comparison results of the 64-point pipelined FFT/IFFT architecture in 4x4 MIMO-OFDM system.

Pipeline Architecture	MIMO-FFT architecture	Complex multipli-cation #	Through-put rate	Complexmultiplier # (Utilization)	Complex adder # (Butterfly Utilization)	Memory Size (Utilization)	Area without memory (Area with memory)
R2SDF [42]	Parallel Multi-Path	98	R	20 (50%)	48 (50%)	252 (100%)	1048 (1375.6)
R2^{2}SDF [34]	Parallel Multi-Path	76	R	8 (75%)	48 (50%)	252 (100%)	448 (775.6)
R2^{3}SDF [31]	Parallel Multi-Path	48	R	8 (87.5%)	48+16T (50%)	252 (100%)	528 (855.6)
R24SDF [68]	Parallel Multi-Path	76	R	8 (75%)	48 (50%)	252 (100%)	448 (775.6)
R4SDF [69]	Parallel Multi-Path	76	R	8 (75%)	96 (25%)	252 (100%)	496 (823.6)
R4SDC [70]	Parallel Multi-Path	76	R	8 (75%)	36 (25%)	504 (100%)	436 (1091.2)
R28SDF [32]	Parallel Multi-Path	48	R	8 (12.5%)	64+8T (25%)	252 (100%)	504 (831.6)
R2MDC [67]	Parallel Multi-Path	98	2R	8 (100%)	24 (100%)	316 (100%)	424 (834.8)
R2^{3}MDC [36]	Parallel Multi-Path	48	2R	8 (87%)	24+8T (100%)	316 (100%)	464 (874.8)
R24MDC [71]	Parallel Multi-Path	76	2R	16 (75%)	56 (71.2%)	380 (100%)	856 (1350)
R4MDC [40]	Serial Blockwise	76	4R	6 (100%)	24 (100%)	348 (100%)	324 (776.4)
Modify R4MDC [61]	Serial Multi-Stream	76	4R	4 (100%)	80+12T (100%)	600 (100%)	340 (1120)
Modify R8MDC [41]	Serial Blockwise	48	5.33R	3.2 (75%)	48+4T (75%)	370 (75%)	228 (709)
Proposed 28MDC	Serial Blockwise	48	4R	3.2 (100%)	32+2T (100%)	320 (100%)	202 (618)

4.5 Summary

This work proposes a hardware-orientated approach for high efficiency to minimize the complex multiplicative complexity, area cost and achieve 100% butterfly utilization with an appropriate throughput rate. By adopting the proposed R8-FFT unit combined with the MAW method, two efficient serial blockwise type 64-point FFT/IFFT processors are constructing for the 2×2 and 4×4 MIMO-OFDM WLAN systems. For the 2×2 MIMO-OFDM system, the proposed R28MDF design has the best performance in terms of lowest complex multiplicative complexity, appropriate throughput rate of 2R, highest butterfly utilization and the fewest complex multipliers, when compared with other existing 64-point FFT/IFFT processor architectures. For the 4×4 MIMO-OFDM system, the proposed R28MDC outperforms existing FFT/IFFT pipeline processor architectures and has the lowest complex multiplicative complexity, an appropriate throughput rate of 4R, highest utilization rate (100%) of all components and the lowest hardware cost. According to the IEEE 802.11n standard [23], execution time for the 128-point and 64-point FFT/IFFT processor with 1–4 simultaneous data sequences must be calculated within 3.6 or 4.0 μs. In total, eight operational modes of the FFT/IFFT processor are required in the IEEE 802.11n standard. The effective reconfigurable FFT/IFFT processor [73] supports eight operational modes in the IEEE 802.11n standard, consumes small hardware and little power, is easily reused, and is an important topic for future work.

Chapter 5 Long-Length based Effective Pipeline FFT/IFFT Processor

In order to demonstrating the high efficiency for the long-length FFT/IFFT computations, the proposed effective architecture focus on the design of 4096-point FFT/IFFT processor ensuring the reasonable operating times for low chip cost and on the features of the high hardware utilization rate. In this chapter, two high effective 4096-point pipeline FFT/IFFT processors have been presented, namely R4^2SDF and R4^3SDF design, to achieve the less complex multiplicative complexities as radix-16 and radix-64 based algorithm with only radix-4 based algorithm. Results of comprehensive comparison further indicate that the proposed R4^2SDF and R4^3SDF based pipeline processors achieve a higher utilization with a smaller hardware requirement than R2^2SDF [31, 34] and other pipeline processors in the 4096-point FFT/IFFT computation, and thus have the higher hardware efficiency. Then, the proposed architectures are very appropriate for the long-length based FFT/IFFT system. The organization of this chapter is structured as follows. A new R4^2SDF and R4^3SDF FFT/IFFT algorithms are given in Section 5.1. Section 5.2 demonstrates the proposed R4^2SDF and R4^3SDF VLSI architectures. The finite word-length analysis is given in Section 5.3, and indicates that the proposed architectures achieve the satisfactory system performance. Section 5.4 tabulates the comparison results in terms of hardware utilization and cost to demonstrate the high cost-efficiency of the proposed architectures. The chip implementation is discussed in Section 5.5. The section 5.6 draws conclusions.

5.1 New Radix-4^2 and Radix-4^3 based FFT/IFFT Algorithm

5.1.1 Radix-4^2 based FFT Formula

The FFT of the N-point input $x[n]$ is given by

$$X[k]=\sum_{n=0}^{N-1} x[n]\cdot W_N^{kn}, \tag{76}$$

where $W_N = e^{-j2\pi/N}$. Applying a 3-dimensional linear index map, the parameters n and k could be expressed as the combinations of n_1, n_2, n_3 and k_1, k_2, k_3, respectively.

$$n=\frac{N}{4}n_1+\frac{N}{16}n_2+n_3, \quad k=k_1+4k_2+16k_3. \tag{77}$$

where $0 \leqq n_1, n_2, k_1, k_2 \leqq 3$.The common factor algorithm (CFA) [64] form can be written as

$$X\left[k_1+4k_2+16k_3\right]=\sum_{n_3=0}^{\frac{N}{16}-1}\sum_{n_2=0}^{3}\sum_{n_1=0}^{3} x\left(\frac{N}{4}n_1+\frac{N}{16}n_2+n_3\right)W_N^{\left(\frac{N}{4}n_1+\frac{N}{16}n_2+n_3\right)(k_1+4k_2+16k_3)}$$

$$=\left\{\sum_{n_3=0}^{\frac{N}{16}-1}\left\{\sum_{n_2=0}^{3} B_{\frac{N}{4}}^{k_1}\left(\frac{N}{16}n_2+n_3\right)W_N^{\frac{N}{16}n_2(k_1+4k_2)}\right\}W_N^{n_3(k_1+4k_2)}\right\}W_{\frac{N}{16}}^{n_3k_3}, \tag{78}$$

where the butterfly structure of the first stage takes the form

$$B_{\frac{N}{4}}^{k_1}\left(\frac{N}{16}n_2+n_3\right)=x\left(\frac{N}{16}n_2+n_3\right)+(-j)^{k_1}x\left(\frac{N}{16}n_2+n_3+\frac{N}{4}\right)$$

$$+(-1)^{k_1}x\left(\frac{N}{16}n_2+n_3+\frac{2N}{4}\right)+(j)^{k_1}x\left(\frac{N}{16}n_2+n_3+\frac{3N}{4}\right), \tag{79}$$

Following a similar decomposition procedure, Eq. (78) can be decomposed as

$$X[k_1+4k_2+16k_3]=\left\{\sum_{n_3=0}^{\frac{N}{16}-1} B_{\frac{N}{16}}^{k_1,k_2}(n_3)W_N^{n_3(k_1+4k_2)}\right\}W_{\frac{N}{16}}^{n_3k_3}, \tag{80}$$

Meanwhile, the butterfly structure of the second stage can be obtained as

$$B_{\frac{N}{16}}^{k_1,k_2}(n_3)=B_{\frac{N}{4}}^{k_1}(n_3)+W_{16}^{k_1}\left[(-j)^{k_2}B_{\frac{N}{4}}^{k_1}\left(n_3+\frac{N}{16}\right)\right]+W_{16}^{2k_1}\left[(-1)^{k_2}B_{\frac{N}{4}}^{k_1}\left(n_3+\frac{2N}{16}\right)\right]+W_{16}^{3k_1}\left[(j)^{k_2}B_{\frac{N}{4}}^{k_1}\left(n_3+\frac{3N}{16}\right)\right], \tag{81}$$

Clearly, the decomposition creates three multipliers: $W_{16}^{k_1}$, $W_{16}^{2k_2}$ and $W_{16}^{3k_3}$, as written in (81). Three full complex multipliers from the second butterfly stage can be simplified as one single constant multiplier in the proposed R4^{2}SDF architecture. The constant multiplier cost can be further reduced by applying the subexpression elimination algorithm. The detailed hardware structure of constant multiplier is described in the next section. The second radix-4 butterfly structure in (81) is the same as the first radix-4 butterfly structure in (79) after simplification of the common factor of the constant multiplier. The complete radix-4^2 decimation-in-frequency (DIF) FFT algorithm is obtained by applying the CFA procedure recursively to the remaining FFTs of length N/16 in (80), as illustrated in Fig. 21. Figure 21 indicate that the proposed radix-4^2 algorithm decomposes the N-points FFT computation by cascading the number of $\log_{16}N$ radix-16 based butterfly (R16-BF) computations, which can be split into two cascading radix-4 based butterfly (R4-BF) computations as depicted in (79) and (81). When the variables of k_1, k_2 and k_3 were treated as constants for each single output $X[k_1 + 4k_2 + 16k_3]$ as depicted in (78) and (80), the summation rages indicate that the required computation results of first and second radix-4 butterfly stage were N/4 and N/16, respectively, as depicted in Fig. 21. The radix-4^2 algorithm has the same multiplicative complexity as the radix-16 algorithm, but still retains the radix-4 butterfly structure. Significantly, the radix-16 algorithm clearly has a lower multiplicative complexity than other low-radix algorithm, such as a radix-2^2 algorithm. For instance, the number of complex multiplications of the 256-point FFT computation adopting the radix-2^2 and radix-4^2 algorithms are 1539 and 224, respectively. Thus, the proposed design based on the new radix-4^2 algorithm has a lower multiplication complexity (85.4%) than the R2^{2}SDF design [31][34]. Furthermore, as mentioned above, the radix-4^2 algorithm does not require any multiplication in the single butterfly structure.

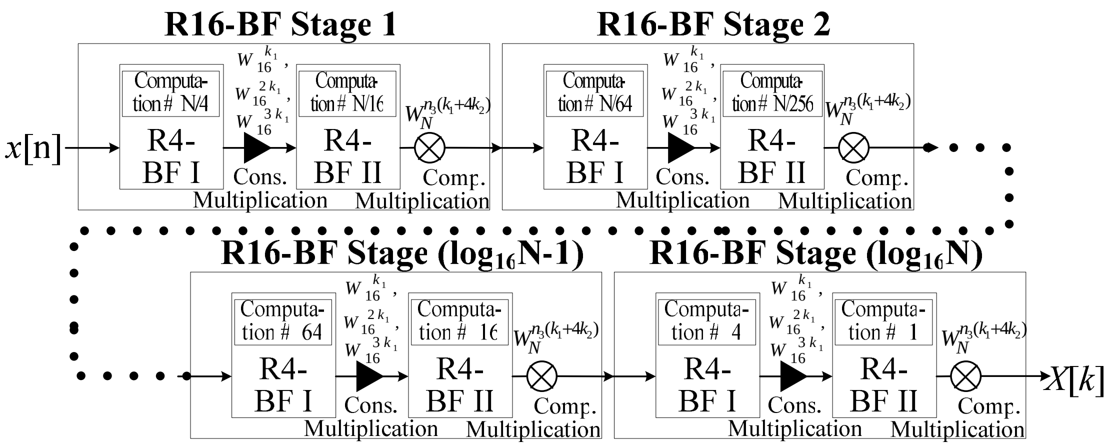


Fig. 21: The CFA decomposition procedure of the proposed radix-4^2 based N-point FFT algorithm.

5.1.2 Radix-4^2 based IFFT Formula

Following the similar procedure, the radix-4^2 IFFT algorithm can be obtained as below. The IFFT of the N-point input $X[k]$ is given by

$$x[n]=\frac{1}{N}\sum_{k=0}^{N-1}X[k]\cdot W_N^{-kn}, \tag{82}$$

In (82), the coefficient $\frac{1}{N}$ can be implemented with the simple right-shift circuit. Thus, the IFFT derivation results can be written as

$$\begin{aligned}
x(n_1+4n_2+16n_3) &= \frac{1}{N}\sum_{k_3=0}^{\frac{N}{16}-1}\sum_{k_2=0}^{3}\sum_{k_1=0}^{3} X\left[\frac{N}{4}k_1+\frac{N}{16}k_2+k_3\right]W_N^{-(\frac{N}{4}k_1+\frac{N}{16}k_2+k_3)(n_1+4n_2+16n_3)} \\
&= \frac{1}{N}\left\{\sum_{k_3=0}^{\frac{N}{16}-1}\left\{\sum_{k_2=0}^{3} B_{\frac{N}{4}}^{n_1}(\frac{N}{16}k_2+k_3)W_N^{-\frac{N}{16}k_2(n_1+4n_2)}\right\}W_N^{-k_3(n_1+4n_2)}\right\}W_{\frac{N}{16}}^{-k_3n_3} \\
&= \frac{1}{N}\left\{\sum_{k_3=0}^{\frac{N}{16}-1} B_{\frac{N}{16}}^{n_1,n_2}(k_3)W_N^{-k_3(n_1+4n_2)}\right\}W_{\frac{N}{16}}^{-k_3n_3},
\end{aligned} \tag{83}$$

where the butterfly structure of the first and second stage has the form

$$B_{\frac{N}{4}}^{n_1}(\frac{N}{16}k_2+k_3)=x(\frac{N}{16}k_2+k_3)+(j)^{n_1}x(\frac{N}{16}k_2+k_3+\frac{N}{4})+(-1)^{n_1}x(\frac{N}{16}k_2+k_3+\frac{2N}{4})+(-j)^{n_1}x(\frac{N}{16}k_2+k_3+\frac{3N}{4})), \tag{84a}$$

and

$$B_{\frac{N}{16}}^{n_1,n_2}(k_3)=B_{\frac{N}{4}}^{n_1}(k_3)+W_{16}^{-n_1}\left[(j)^{n_2}B_{\frac{N}{4}}^{n_1}(k_3+\frac{N}{16})\right]+W_{16}^{-2n_1}\left[(-1)^{n_2}B_{\frac{N}{4}}^{n_1}(k_3+\frac{2N}{16})\right]+W_{16}^{-3n_1}\left[(j)^{n_2}B_{\frac{N}{4}}^{n_1}(k_3+\frac{3N}{16})\right]. \tag{84b}$$

Notably, the only difference between FFT and IFFT algorithm are the sign bits as given in (79), (81) (84a) and (84b). Therefore, the pipeline FFT/IFFT processor can be easily implemented with a single module by controlling the sign coefficient. Additionally, the proposed pipeline IFFT processor has a similar butterfly structure and a single constant multiplier structure with the proposed pipeline FFT processor, which could replace the three multipliers: $W_{16}^{-n_1}$, $W_{16}^{-2n_2}$ and $W_{16}^{-3n_3}$.

5.1.3 Radix-4^3 based FFT/IFFT Formula

Applying another 4-dimensional linear index map in (76), the parameters n and k could be expressed as the combinations of n_1, n_2, n_3, n_4 and k_1, k_2, k_3, k_4, respectively.

$$n = \frac{N}{4}n_1 + \frac{N}{16}n_2 + \frac{N}{64}n_3 + n_4,$$

$$k = k_1 + 4k_2 + 16k_3 + 64k_4. \tag{85}$$

where 0 ≦ n_1, n_2, n_3, k_1, k_2, k_3 ≦ 3. The common factor algorithm (CFA) [64] form can be written as

$$X[k_1 + 4k_2 + 16k_3 + 64k_4] = \sum_{n_4=0}^{\frac{N}{64}-1}\left\{\sum_{n_3=0}^{3}\left\{\sum_{n_2=0}^{3} B_{\frac{N}{4}}^{k_1}\left(\frac{N}{16}n_2 + \frac{N}{64}n_3 + n_4\right)\right.\right.$$
$$\left.\left.\cdot W_N^{\frac{N}{16}n_2(k_1+4k_2)}\right\} W_N^{\frac{N}{64}n_3(k_1+4k_2+16k_3)}\right\} W_N^{n_4(k_1+4k_2+16k_3+64k_4)}$$

$$= \sum_{n_4=0}^{\frac{N}{64}-1}\left\{\sum_{n_3=0}^{3} B_{\frac{N}{16}}^{k_1,k_2}\left(\frac{N}{64}n_3 + n_4\right) W_N^{\frac{N}{64}n_3(k_1+4k_2+16k_3)}\right\}\cdot W_N^{n_4(k_1+4k_2+16k_3+64k_4)}$$

$$= \sum_{n_4=0}^{\frac{N}{64}-1}\left\{B_{\frac{N}{64}}^{k_1,k_2,k_3}(n_4)\cdot W_N^{n_4(k_1+4k_2+16k_3)}\right\}\cdot W_{\frac{N}{64}}^{n_4k_4}, \tag{86}$$

where the butterfly structure of the each stage takes the form

The first butterfly stage:

$$B^{k_1}_{\frac{N}{4}}(\frac{N}{16}n_2+\frac{N}{64}n_3+n_4)=x(\frac{N}{16}n_2+\frac{N}{64}n_3+n_4)+(-j)^{k_1}x(\frac{N}{16}n_2+\frac{N}{64}n_3+n_4+\frac{N}{4})$$
$$+(-1)^{k_1}x(\frac{N}{16}n_2+\frac{N}{64}n_3+n_4+\frac{2N}{4})+(j)^{k_1}x(\frac{N}{16}n_2+\frac{N}{64}n_3+n_4+\frac{3N}{4})$$

The second butterfly stage:

$$B^{k_1,k_2}_{\frac{N}{16}}(\frac{N}{64}n_3+n_4)=B^{k_1}_{\frac{N}{4}}(\frac{N}{64}n_3+n_4)+W_{16}^{k_1}\left[(-j)^{k_2}B^{k_1}_{\frac{N}{4}}(\frac{N}{64}n_3+n_4+\frac{N}{16})\right]$$
$$+W_{16}^{2k_1}\left[(-1)^{k_2}B^{k_1}_{\frac{N}{4}}(\frac{N}{64}n_3+n_4+\frac{2N}{16})\right]+W_{16}^{3k_1}\left[(j)^{k_2}B^{k_1}_{\frac{N}{4}}(\frac{N}{64}n_3+n_4+\frac{3N}{16})\right]$$

The third butterfly stage:

$$B^{k_1,k_2,k_3}_{\frac{N}{64}}(n_4)=B^{k_1}_{\frac{N}{16}}(n_4)+W_{64}^{(k_1+4k_2)}\left[(-j)^{k_3}B^{k_1}_{\frac{N}{16}}(n_4+\frac{N}{64})\right]+W_{64}^{2(k_1+4k_2)}$$
$$\left[(-1)^{k_3}B^{k_1}_{\frac{N}{16}}(n_4+\frac{2N}{64})\right]+W_{64}^{3(k_1+4k_2)}\left[(j)^{k_3}B^{k_1}_{\frac{N}{16}}(n_4+\frac{3N}{64})\right], \tag{87}$$

The complete radix-4^3 DIF FFT algorithm is obtained by applying the CFA procedure recursively to the remaining FFTs of length N/64 in (86). Thus, the radix-4^3 algorithm has few multiplicative complexities as the radix-64 algorithm, but still retains the simple radix-4 butterfly structure. For instance, the numbers of complex multiplications in the 4096-point FFT computation adopting the radix-2^2, radix-4^2 and radix-4^3 algorithms are 13996, 7425 and 3969, respectively. Thus, the proposed radix-4^3 algorithm has a lower multiplication complexity (71.6%) than the radix-2^2 algorithm [31, 34]. Significantly, the radix-4^3 algorithm clearly has a lower multiplicative complexity than the purposed radix-4^2 algorithm and other low-radix algorithms. According to the similar radix-4 based butterfly architecture with only some sign inversions, the radix-4^3 DIF IFFT computation could be obtained.

5.2 Pipeline 4096-Point R4^{2}SDF and R4^{3}SDF based FFT/IFFT VLSI Architecture

Base on the new proposed radix-4^2 and radix-4^3 DIF FFT algorithms, the novel R4^{2}SDF and R4^{3}SDF architectures for supporting the 4096-point FFT/IFFT computations are shown in Fig. 22 and 23, respectively. Two proposed architectures both require six butterfly stages with 4095-word shift registers. The R4^{2}SDF based 4096-point FFT/IFFT pipeline processor requires three constant multipliers and two complex multipliers. The R4^{3}SDF based 4096-point FFT/IFFT pipeline processor requires four constant multipliers and one complex multiplier. Comparing with the R4^{2}SDF design, the R4^{3}SDF design replaces one complex multiplier with one constant multiplier in the 4096-point FFT/IFFT computation. The detailed operations of each element are described as follows.

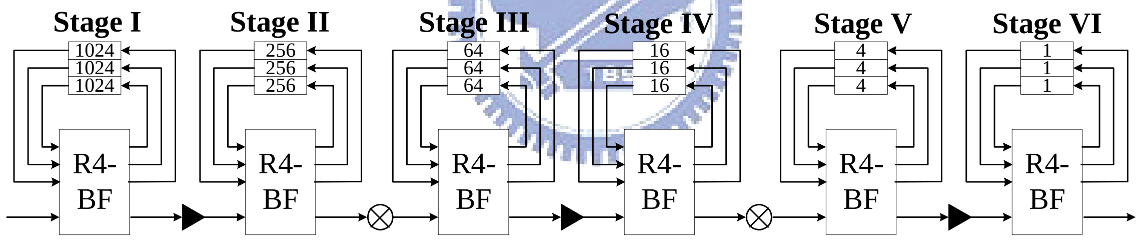


Fig. 22: Block diagram of the R4^{2}SDF-based 4096-point FFT/IFFT VLSI architecture.

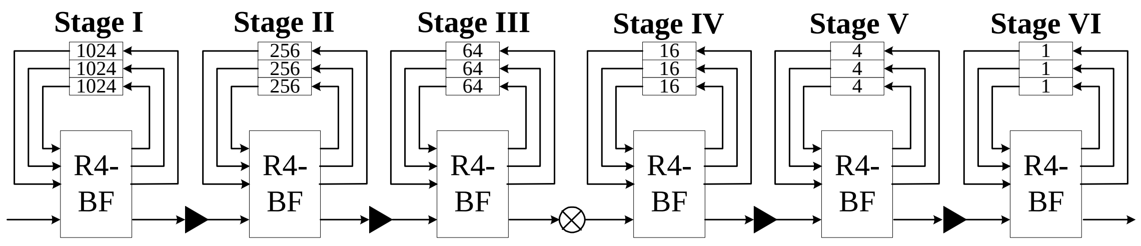


Fig. 23: Block diagram of the R4^{3}SDF-based 4096-point FFT/IFFT VLSI architecture.

5.2.1 Radix-4 Butterfly

The derivation results of the radix-4^2 and radix-4^3 algorithms reveal that both the FFT/IFFT butterfly computation in (78) and (86), can be easily computed with the same radix-4 butterfly architecture. Notably, the radix-4 butterfly structure only requires trivial multiplication, which involves real-imaginary swapping and sign inversion, and which does not require any complex multiplication. Figure 24 illustrates the proposed radix-4 butterfly structure, which only includes four four-input complex adders. Without any complex multiplier, the radix-4 based butterfly structure is more cost-efficient than higher-radix based butterfly structures. Moreover, the proposed radix-4^2 algorithm has the same complex multiplication complexity as the radix-16 algorithm, and radix-4^3 algorithm further has the few complex multiplication complexity as the radix-64 algorithm. Thus, the proposed two pipeline architectures have the high cost efficiency of lower radix architectures.

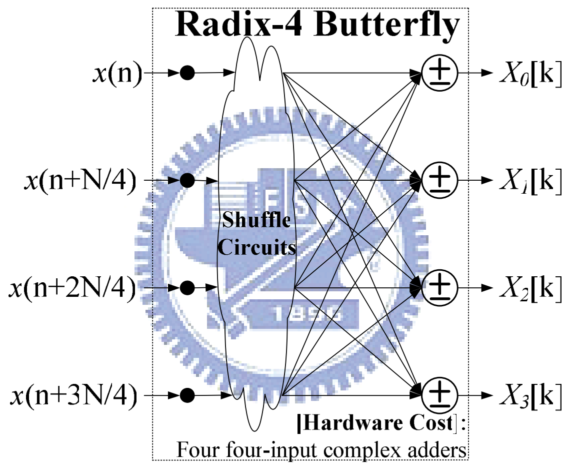
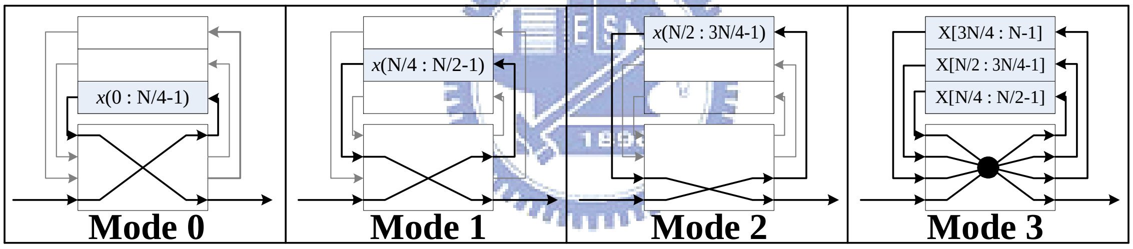


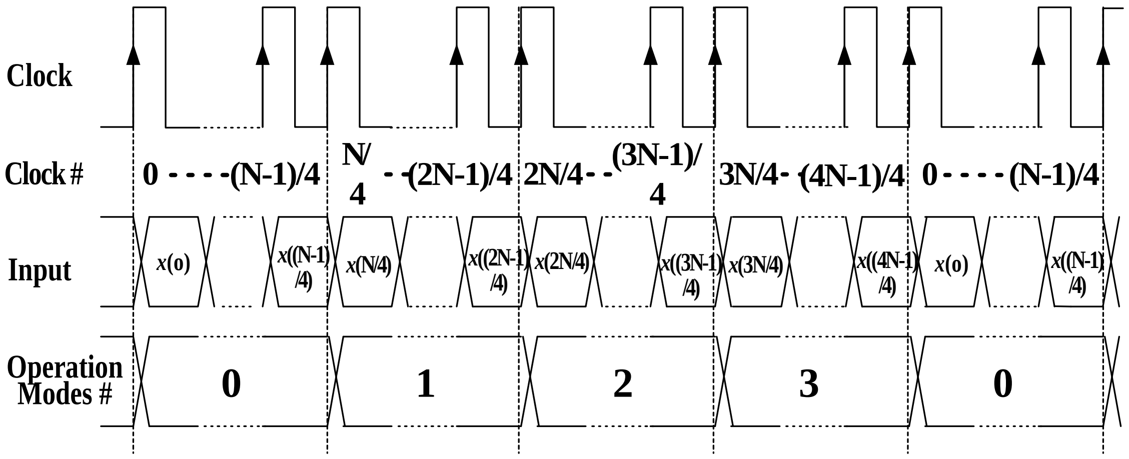
Fig. 24: Block diagram of the radix-4 butterfly architecture.

5.2.2 Memory Structure

The memory structure of each butterfly stage is well known to be an important issue for the effective pipeline FFT/IFFT processor. In this work, the delay feedback based memory structure is adopted. In order to compute the radix-4 based butterfly computations, the input data and the intermediate results have to be reordered as four concurrently data streams using memory as shown in Fig. 24. In the radix-4 butterfly structure, four proposed operation modes can finish the data reordering and the butterfly computation as shown in Fig. 25(a). Operation modes 0–2 are adopted in the data reordering, and operation mode 3 is adopted in the FFT/IFFT computation. Each radix-4 butterfly unit applies three parallel Fist-In First-Out (FIFO) shift registers to store the serial data input and butterfly output in the feedback paths as presented in Fig. 25(a). The timing sequence of N-point FFT/IFFT computation can be divided into four stages, each stage contains N/4 clock cycles as presented in Fig. 25(b). The required number of memory cells for the *k*th stage is $3\times N/(4^k)$. Significantly, the SDF based pipeline FFT/IFFT structure is highly regular, which has the highly effective memory structure with the simpler routing complexity [31, 32, 34, 35, 42, 43].



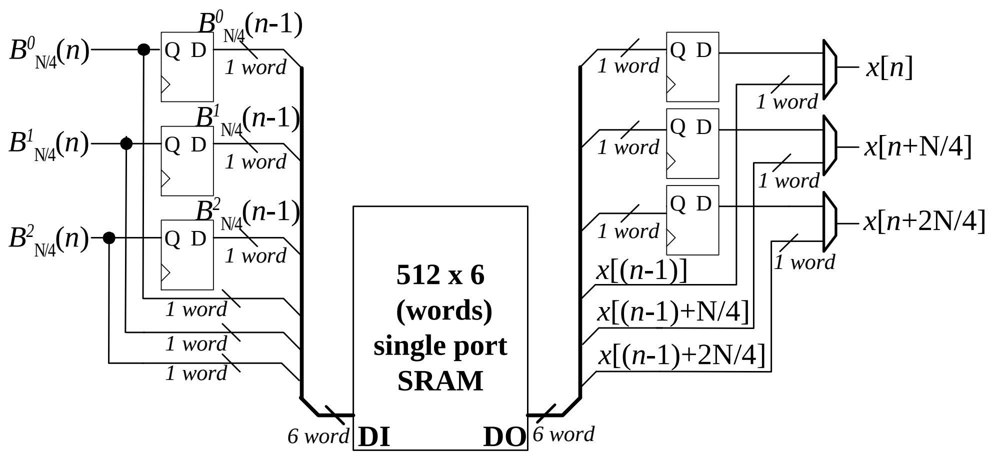
(a) The proposed 4 operation modes in the radix-4 based butterfly stages.



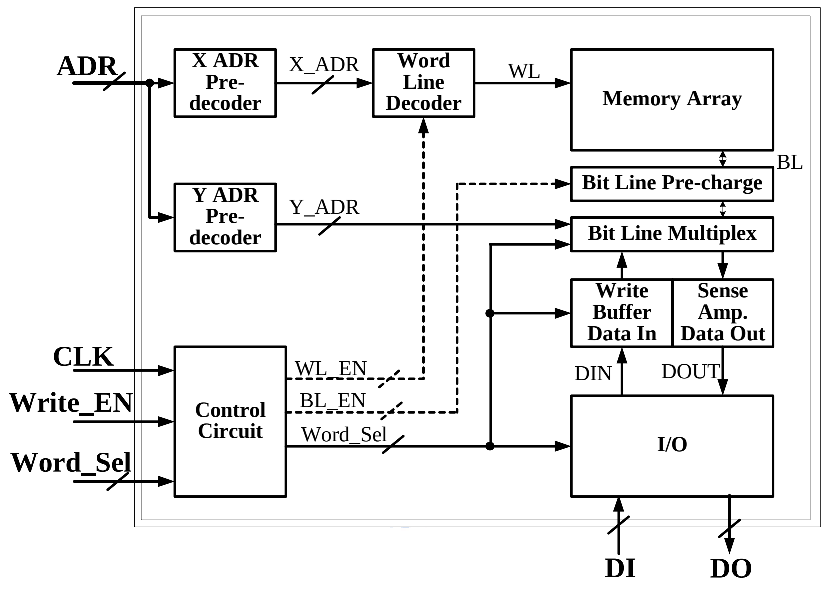
(b) The timing sequences of 4 operation modes in the proposed pipeline architecture.

Fig. 25: The proposed 4 operation modes of the radix-4 butterfly stage in the R4^{2}SDF and R4^{3}SDF based 4096-point FFT/IFFT VLSI architecture.

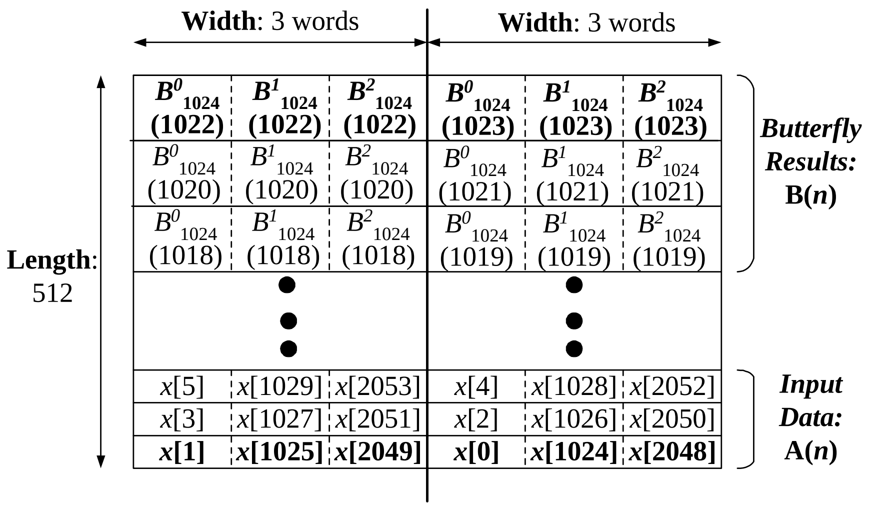
The dual port memory is well known to be an intuitive implementation for the FIFO shifts register. However, each cell in the dual port memory takes an area 33% larger than the corresponding single port RAM cell. Furthermore, the dual port memory would consume more power than single port memory [31]. In this study, the memory implementation of stage I and II are realized by the single port SRAM. The proposed FIFO shift registers architecture in the butterfly stage I is depicted in Fig. 26(a), where the notations of the input/output ports denote the respective operators in (79) for the proposed operation mode 3. Due to the few memory cell requirements, the stage III, IV, V and VI adopt the synchronize flip-flops to implement the FIFO shift registers for the small chip cost. Accompany with the six words synchronize flip-flops, the proposed FIFO architecture has a wide data width of six-words to provide a six-words reading at a time as shown in Fig. 26(a). Base on the proposed FIFO shifter registers as depicted in Fig. 26(a), the proposed memory architecture can concurrently provide three operators for the radix-4 based butterfly unit in the current and consequent cycles. Therefore, the size of single port SRAMs are 512×6 and 128×6 words in the stage I and II, respectively. Accompany with the control signals of word selection, the proposed single port SRAM adopts the simple word-control circuits to provide the ability of independent-word writing in the same address as shown in Fig. 26(b). That means the proposed single port SRAM, which has the wide data width, can easily achieve the independent-word writing for the data reordering in the operation modes 0–2 as shown in Fig. 25(a). The detail data arrangement in the proposed single port memory is listed as Fig. 26(c). In Fig. 26(c), the notation A(n) and B(n) denote the combinative data sets of three input data and butterfly results after data reordering and butterfly computations, respectively. In the butterfly stage I, A(n) and B(n) could be expressed as {$x(n)$, $x(n$+N/4), $x(n$+N/2)} and {$B^0_{\mathrm{N/4}}(n)$, $B^1_{\mathrm{N/4}}(n)$, $B^2_{\mathrm{N/4}}(n)$}, respectively. Notably, each radix-4 butterfly unit could store the input data and output results in the same SRAM for the highest memory utilization rate. The read and write operations are interleaved and each of them is active every other clock cycle as shown in Fig. 26(d), which can prevent the read/write conflict. Figure 26(d) shows the detail timing sequence of the proposed memory architecture in the operation mode 3.



(a) The proposed FIFO shift registers architecture on the butterfly stage I.



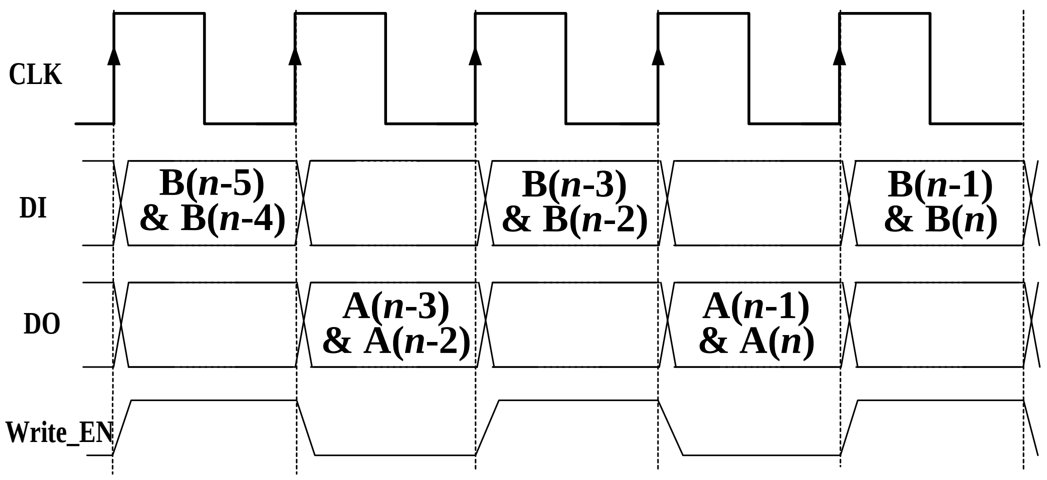
(b) The proposed single port SRAM with independent word control.



Width: 3 words
Width: 3 words
Length: 512
Butterfly Results: **B(*n*)**
Input Data: **A(*n*)**

$\boldsymbol{B^0_{1024}}$ **(1022)**	$\boldsymbol{B^1_{1024}}$ **(1022)**	$\boldsymbol{B^2_{1024}}$ **(1022)**	$\boldsymbol{B^0_{1024}}$ **(1023)**	$\boldsymbol{B^1_{1024}}$ **(1023)**	$\boldsymbol{B^2_{1024}}$ **(1023)**
B^0_{1024} (1020)	B^1_{1024} (1020)	B^2_{1024} (1020)	B^0_{1024} (1021)	B^1_{1024} (1021)	B^2_{1024} (1021)
B^0_{1024} (1018)	B^1_{1024} (1018)	B^2_{1024} (1018)	B^0_{1024} (1019)	B^1_{1024} (1019)	B^2_{1024} (1019)
● ● ●			● ● ●		
x[5]	*x*[1029]	*x*[2053]	*x*[4]	*x*[1028]	*x*[2052]
x[3]	*x*[1027]	*x*[2051]	*x*[2]	*x*[1026]	*x*[2050]
***x*[1]**	***x*[1025]**	***x*[2049]**	***x*[0]**	***x*[1024]**	***x*[2048]**

(c) The memory context on the purposed butterfly stage I.



(d) The timing sequence of proposed memory architecture in the operation mode 3.

Fig. 26: The proposed memory architecture of the butterfly stage I and II in the R4^{2}SDF and R4^{3}SDF based 4096-point FFT/IFFT VLSI architecture.

5.2.3 Constant Multiplier

Based on the derivation results in Section 5.1, the radix-4^2 algorithm requires some complex multiplications, namely $W_{16}^{k_1}$, $W_{16}^{2k_1}$ and $W_{16}^{3k_1}$ in the 4096-point FFT/IFFT computation in (81). According to the SDF based architecture as depicted in Fig. 22, a single data stream passes through the constant multipliers and complex multipliers. There is only one complex multiplication, which is computed in (81) during each cycle. Then, the three full complex multipliers can be simplified as a single constant multiplier. This subsection follows three steps to reduce the complex multipliers to the most economical constant multipliers in the R4^{2}SDF and R4^{3}SDF architecture. The implementation of constant multiplier in the R4^{2}SDF architecture is presented as below. First, the multiplication of twiddle factors from Eq. (81) is realized as the constant multiplier, which only contains shifters and adders as shown in Fig. 27. Second, the complex conjugate symmetry rule is applied to decrease the number of complex multiplications to only two constant multiplications per stage with some shuffle circuits as shown in Fig. 27, thus achieving a constant multiplier cost reduction of 83%. Finally, the subexpression elimination algorithm [65] is adopted to reduce the number of shift circuits by more than 20%, and the number of complex adders by 50% in one constant multiplier, as depicted in Fig. 27. The strictest constant multipliers are obtained in the purposed architectures by following these three steps. The cost penalty of the constant multiplier is thus minimized. Similarity, the radix-4^3 algorithm has two retrenched constant multipliers as depicted in (78). The constant multiplier of second stage in R4^3SDF design is the same as the constant multiplier in R4^2SDF design. Following the similar reduction steps, the constant multiplier of the third stages in R4^3SDF based design requires eight constant multiplications with the cost reduction of 83%. Considering the chip cost in R4^3SDF design, the constant multiplier in third stage increases slightly control complexity than the constant multiplier in second stage.

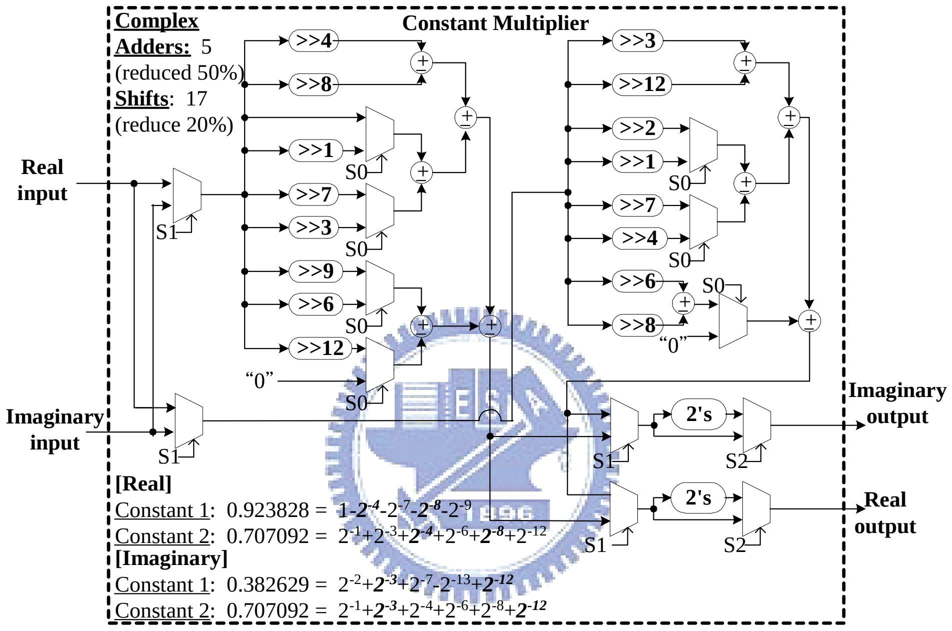

Complex Adders: 5 (reduced 50%)
Shifts: 17 (reduce 20%)
Constant Multiplier
Real input
Imaginary input
>>4
>>8
>>1
>>7
>>3
>>9
>>6
>>12
>>2
“0”
S0
S1
S2
2's
Imaginary output
Real output
[Real]
Constant 1: 0.923828 = $1-2^{-4}-2^{-7}-2^{-8}-2^{-9}$
Constant 2: 0.707092 = $2^{-1}+2^{-3}+2^{-4}+2^{-6}+2^{-8}+2^{-12}$
[Imaginary]
Constant 1: 0.382629 = $2^{-2}+2^{-3}+2^{-7}-2^{-13}+2^{-12}$
Constant 2: 0.707092 = $2^{-1}+2^{-3}+2^{-4}+2^{-6}+2^{-8}+2^{-12}$

Fig. 27: Block diagram of the proposed constant multiplier in R4^2SDF design.

5.2.4 Eight-Folded Complex Multiplier

The proposed 4096-point R4^{3}SDF design has only one complex multiplier and one coefficient ROM to realize the complex multiplication of twiddle factors $W_N^{n_4(k_1+4k_2+16k_3)}$ in (86). However, the proposed 4096-point R4^{2}SDF design requires two complex multipliers and two coefficient ROMs to realize the $W_N^{n_3(k_1+4k_2)}$ in (79). To decrease the ROM size, the complex conjugate symmetry rule and subexpression elimination [65] is applied to devise one eight-folded complex multiplier as shown in Fig. 28. The proposed eight-folded complex multiplier could reduce the storage size of 87.5 % for each coefficient ROM. In the proposed R4^{2}SDF design, the first and second coefficient ROMs store 31 and 511 words, respectively. However, the proposed R4^{3}SDF design only has one complex multiplier, which stores 511 words in the coefficient ROM. Comparing with the R4^{3}SDF design, the R4^{2}SDF design requires a larger chip cost of two complex multipliers and two coefficient ROMs to complete the 4096-point FFT/IFFT computation. The ROM address and data control circuit of R4^{3}SDF design are easily realized by the 12-bit counter controller given in Table 6.

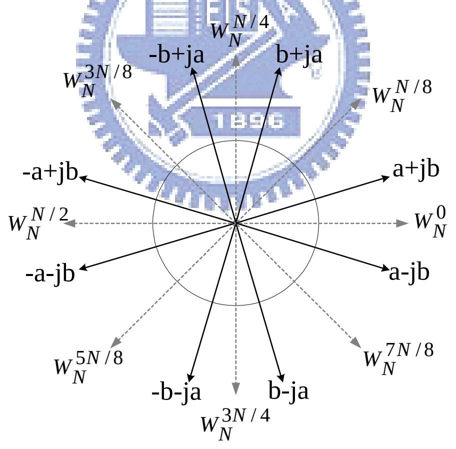


Fig. 28: The block diagram of eight-folded algorithm in the coefficient ROM.

Table 7: The Data Control of The Coefficient ROM in the R4^{3}SDF design.

H = $n_3(k_1+4k_2)$	**Address Mode (H[9])**	**ROM address**	**Data Mode (H[7:5])**	**ROM data**
0~511	0	Two's complement of H[9:0]	0	a+jb
512~1023	1	H[9:0]	1	b+ja
1024~1535	0	Two's complement of H[9:0]	2	-b+ja
1536~2047	1	H[9:0]	3	-a+jb
2048~2559	0	Two's complement of H[9:0]	4	-a-jb
2560~3071	1	H[9:0]	5	-b-ja
3072~3583	0	Two's complement of H[9:0]	6	b-ja
3584~4095	1	H[9:0]	7	a-jb

5.3 Finite Word-Length Analysis

Due to the requirements of handheld devices, several specific issues should be considered — small dimensions, light weight, and battery-power operation. The system performance should then satisfy the relative specifications. A higher system performance undoubtedly implies a larger chip cost and greater power consumption, owing to the wider internal word-length. Since the chip cost and system performance are known to be a trade-off, this study performed a finite word-length analysis to estimate the appropriate word-length for the R4^{2}SDF and R4^{3}SDF based 4096-point FFT/IFFT processors. In this work, the output signal to noise ratio (SNR) performance of 4096-point FFT/IFFT processor is estimated under 40dB additive white Gaussian noise (AWGN) channel. In our fixed-point simulation environment, the input data of the double floating-point precision were generated from the ideal IFFT (FFT) model by passing the 40 dB AWGN channel model in Matlab. The input data with noise are sent into the proposed R4^{2}SDF and R4^{3}SDF pipeline FFT/IFFT architectures, which are modeled at different fixed-point levels for each function unit. The output SNR is obtained by comparing the original input data with the fixed-point model output. The results after 100,000 iterations are averaged as depicted in Fig. 29, where the *x*-axis and *y*-axis represent the internal word-length and the whole system output SNR, respectively. These analytical results demonstrate that the output SNR saturated as the internal word-length increased. It is obviously that the proposed R4^{3}SDF only requires 13-bit internal word-length for each function unit to produce satisfactory performance under 40dB noise environments, satisfying the DVB-H specification [27, 28]. Significantly, the proposed R4^{2}SDF requires one more bit than R4^{3}SDF, which is 14-bit internal word-length for each function units. That means the R4^{2}SDF design has the larger chip cost than R4^{3}SDF design in the 4096-point FFT/IFFT computation.

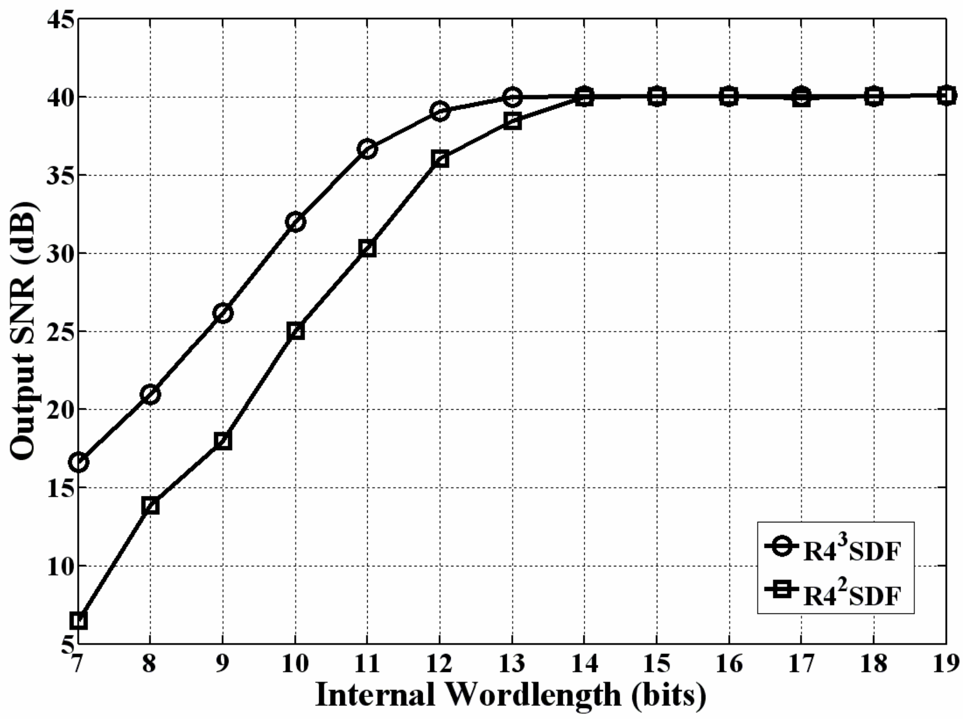


Fig. 29: Finite word-length analysis of the proposed pipeline R4^{2}SDF and R4^{3}SDF-based 4096 points FFT/IFFT architecture.

5.4 The Comparison of Pipeline FFT/IFFT Architecture

This section presents the comprehensive comparison results of several famous pipeline FFT/IFFT architectures to demonstrate the high efficiency of the proposed R4^2SDF and R4^3SDF FFT/IFFT architectures. The architectures are compared in two indices, namely cost and utilization, to express the hardware efficiency of the proposed FFT/IFFT architecture, as listed in Tables 8 and 9. Table 8 lists the required hardware resources, where T denotes the number of complex adders required in the implementation of the constant multiplier. Significantly, the area of the complex multiplier and memory are well known to be the dominant cost index in the pipeline FFT/IFFT design. The comparison results in Table 8 clearly demonstrate that the proposed R4^3SDF based-FFT/IFFT architecture has the fewest complex multipliers requirement among other pipeline architectures. The R4^3SDF based 4096-point FFT/IFFT architecture only needs one complex multiplier, which is 80% and 95% below the requirement of the R2^2SDF and R8MDC FFT/IFFT architectures, respectively. Additionally, the proposed architectures maintain the minimum shift registers requirement among the tested pipeline architectures. Although the proposed R4^2SDF and R4^3SDF based architectures need slightly more complex adders than the R2^2SDF based architecture, this small cost penalty is acceptable. To estimate the total chip cost in the 4096-point FFT/IFFT architectures, which includes the number of complex multipliers, complex adders and memory size, the conventional comparative methodology [26, 34] with the unit of equivalent adders was used to estimate the cost value between the different architectures. Based on the implementation results in our process, we convert the area of each complex multiplier and complex memory to the 50 and 1.3 complex adder, respectively, and the scheme with three real multiplications and five real additions, in the complex multiplier implementation. The rightmost column of Table 8 lists the area indexes of the equivalent adder of the 4096-point FFT/IFFT architecture. Clearly, the proposed R4^3SDF-based 4096-point FFT/IFFT architecture has the lowest hardware requirements. Significantly, the cost advantages of our proposed architectures become more evident when the transform length is larger. That means the proposed architectures are very appropriate for the long-length FFT/IFFT computation. Thus, the proposed R4^3SDF architectures have lower hardware cost than R4^2SDF and other famous pipeline FFT/IFFT architectures in terms of the number of ROMs, complex multipliers, complex adders, constant multipliers and shift registers.

Table 8: Hardware Cost Comparisons of the Pipelined FFT/IFFT Architecture.

Pipeline architecture	Mult. Comp-lexity	Complex Mult.	Complex adders (including constant mult.)	Complex Memory Size	Equivalent area in 4096 points
R2SDF [17]	Radix-2	$\log_2 N-2$	$2\log_2 N$	N-1	5847.5
R4SDF [18]	Radix-4	$\log_4 N-1$	$8\log_4 N$	N-1	5621.5
R8SDF [8]	Radix-8	$\log_8 N-1$	$(24+2T)\log_8 N$	N-1	5609.5
$R2^2$SDF [6]	Radix-2^2	$\log_4 N-1$	$4\log_4 N$	N-1	5597.5
$R2^3$SDF [5]	Radix-2^3	$2(\log_8 N-1)$	$6\log_8 N$	N-1	5647.5
R2MDC [13]	Radix-2	$\log_2 N-2$	$2\log_2 N$	1.5N-2	8508.6
$R2^2$MDC [9]	Radix-2^2	$\log_2 N-2$	$2\log_2 N$	1.5N-2	8508.6
R4MDC [14]	Radix-4	$3\log_4 N-3$	$4\log_2 N$	2.5N-4	14104.8
R8MDC [15]	Radix-8	$7\log_8 N-7$	$(24+2T)\log_8 N$	4.5N-8	30664.4
Proposed $R4^2$SDF	Radix-4^2	$\log_{16} N-1$	$(16+T)\log_{16} N$	N-1	5470.5
Proposed $R4^3$SDF	Radix-4^3	$\log_{64} N-1$	$(24+2T)\log_{64} N$	N-1	5429.5

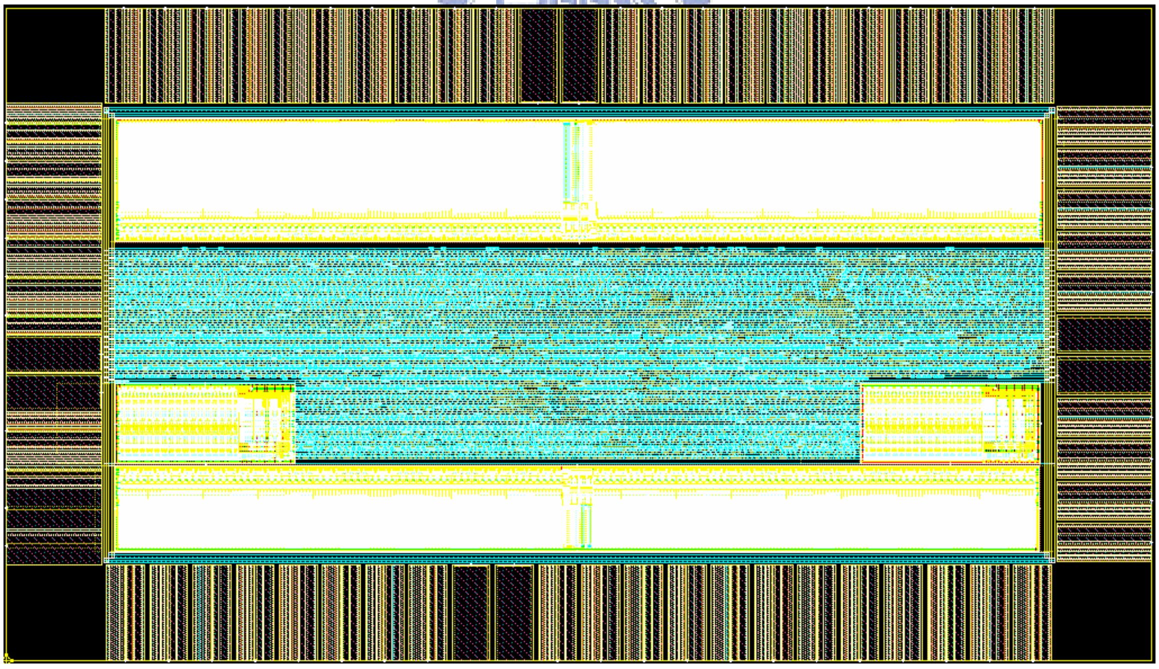
Table 9: Hardware Utilization Rate Comparisons of the Pipelined FFT/IFFT Architecture.

Pipeline architecture	Utilization rate of complex Mult.	Utilization rate of complex adders (including constant mult.)	Utilization rate of complex memory
R2SDF [17]	50%	50%	100%
R4SDF [18]	75%	25%	100%
R8SDF [8]	87.5%	12.5%	100%
$R2^2$SDF [6]	75%	50%	100%
$R2^3$SDF [5]	87.5%	50%	100%
R2MDC [13]	50%	50%	50%
$R2^2$MDC [9]	37.5%	50%	50%
R4MDC [14]	25%	25%	25%
R8MDC [15]	12.5%	12.5%	12.5%
Proposed $R4^2$SDF	87.5%	56.25%	100%
Proposed $R4^3$SDF	96.9%	60.42%	100%

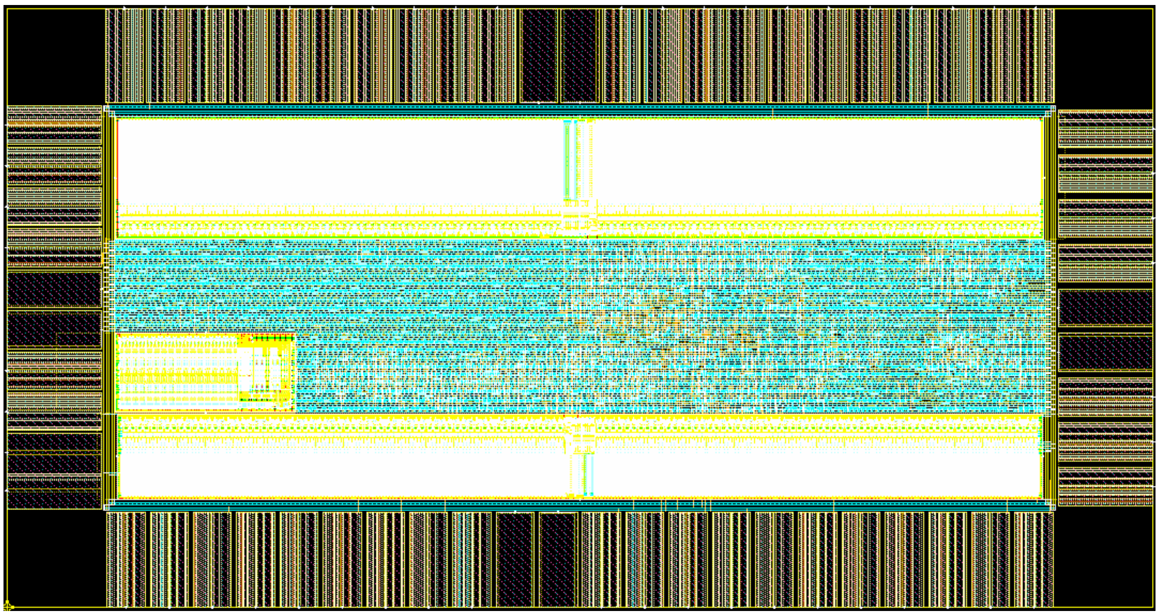
Table 9 shows the comprehensive comparison of the hardware utilization rate in terms of the utilization rate of complex multipliers, complex adders and complex memory. Clearly, the proposed R4^{3}SDF architecture achieves the highest complex multiplier utilization rate among the tested pipeline architectures (96.9%). Additionally, the proposed architecture maintains the maximum complex memory utilization rate of 100%. Furthermore, the proposed R4^{3}SDF architecture, including the constant multipliers, has the highest complex adder utilization rate of 60.42%. Thus, the purposed R4^{3}SDF architecture achieves a higher hardware utilization rate than R4^{2}SDF and other well-known pipeline FFT/IFFT architectures in terms of the utilization rate of complex multipliers, complex adders, constant multipliers and complex memory.

5.5 Chip Implementation

Following the functional verification in the Matlab environment, the proposed R4^{2}SDF and R4^{3}SDF based 4096-point FFT/IFFT architectures in which the internal word-length of entire design are 14-bit and 13-bit, respectively, were synthesized by the Design Compiler with TSMC 0.13μm CMOS technology. Using the standard logic process rules, the single port SRAM applies the 6T bit cell. The floorplan and post-layout were performed by Astro. The post-simulation was issued by NC-Simulator to verify the functionality after back-annotation was performed from the Start-RC extractor. The static timing check can be signed-off by PrimeTime. Finally, the power analysis and DRC were conducted using Astro Rail and Dracula, respectively. The core area of the post layout for the R4^{2}SDF and R4^{3}SDF design are 1.01 and 0.89 mm^2, which includes power rings and power straps as depicted in Fig. 30(a) and 30(b) , respectively. The gate count usage of each building block for R4^{2}SDF and R4^{3}SDF design are listed in Table 10. Comparing with the R4^{2}SDF architecture, the R4^{3}SDF architecture can replace one complex multiplier with one constant multiplier in the 4096-point FFT/IFFT computation as depicted in Fig. 22 and 23. Then, the R4^{3}SDF design reduces the multiplier cost of 3.9 % than the R4^{2}SDF design as listed in Table 10, which includes the complex and constant multipliers. It is obviously that 4095 words feedback memory dominates the chip of 77.34 % and 80.72 % for the R4^{2}SDF and R4^{3}SDF design, respectively. Both of these two chips could operate at 20 MHz, thus satisfying the high throughput requirement. Concerning the speed performance, because the pipelined multiplier operation is easy to design at a clock rate of 20 MHz or even higher, the proposed architectures can achieve a high clock rate by simple pipelining techniques for the involved arithmetic components. The average power dissipation of the R4^{2}SDF and R4^{3}SDF based 4096-point FFT/IFFT design are 6.3725 and 5.985 mW@20 MHz at 1.2V supply voltage. The layout view of R4^{2}SDF design as shown in Fig. 30(a) has 68 I/O pins, of which eight pins are power supply pins. Due to the few datawidth requirements, the layout view of R4^{3}SDF design as shown in Fig. 30(b) has only 64 I/O pins. The proposed R4^{2}SDF and R4^{3}SDF based 4096-point FFT/IFFT implementation satisfies the system performance of DVB-H standard. Additionally, the proposed R4^{3}SDF based 4096-point FFT/IFFT implementation has a low power consumption (5.985 mW), and the lowest hardware requirement among the tested pipeline architectures. These findings indicate that the proposed design meets the requirements of high effective pipeline FFT/IFFT processor for SoC IP.



(a) The layout view of proposed R4^{2}SDF design.



(b) The layout view of proposed R4^{3}SDF design.

Fig. 30: The layout view of proposed 4096-point pipeline FFT/IFFT processor.

Table 10: The Gate Count Usage of Each Building Block in the Proposed Design.

Categories	Control	Butterfly Cores	Complex Multiplier	Constant Multipliers	Shift Registers
R4^{2}SDF	0.33 %	10.1 %	9.83 %	2.4 %	77.34 %
R4^{3}SDF	0.35 %	10.6 %	5.03 %	3.3 %	80.72 %

5.6 Summary

This work develops two high effective R4^{2}SDF and R4^{3}SDF pipeline VLSI architectures that support the long-length FFT/IFFT computations. The proposed R4^{3}SDF pipeline FFT/IFFT architecture has lower multiplicative complexity and higher hardware utilization rate with smaller cost than R4^{2}SDF and other pipeline architectures. Following with fixed-point analysis in 40dB AWGN environment, the proposed R4^{2}SDF and R4^{3}SDF based 4096-point FFT/IFFT designs are successfully implemented in 0.13 μm CMOS technology with an internal word-length of 14 and 13-bits, respectively. The proposed R4^{2}SDF and R4^{3}SDF based design have a low power consumption of 6.3725 and 5.985 mW @20 MHz at 1.2V supply voltage. Thus, these features ensure that the proposed R4^{3}SDF pipeline 4096-points FFT/IFFT processor design certainly meets the high effective VLSI architecture.

Chapter 6 Effective Triple-Mode Reconfigurable Pipeline FFT/IFFT/2D-DCT Processor

Tell *et al.* [8] presented the FFT/WALSH/1-D DCT processor for multiple radio standards of the upcoming 4^{th} generation wireless systems. Conversely, some designs [8-10] only support 1-D DCT computation, and have no 2-D DCT support. However, 2-D DCT is desirable for the video compression among wireless communication applications. This study not only presents a single reconfigurable architecture for the 256-point FFT/IFFT modes and the 8×8 2-D DCT mode, but also achieves high cost-efficiency in portable multimedia applications. Results of comprehensive comparison further indicate that the proposed $R4^2SDF$-based pipeline processor achieves a higher utilization with a smaller hardware requirement than $R2^2SDF$-based pipeline processor [31] in the 256-point FFT/IFFT mode, and thus has higher cost efficiency. The proposed $R4^2SDF$-based design also achieves satisfactory performance for the DV encoding standard with the lowest cost in the 8×8 2D DCT mode. The organization of this chapter is structured as follows. A new $R4^2SDF$ FFT/IFFT and 8×8 2D DCT algorithm is given in Section 6.1. Section 6.2 demonstrates the proposed FFT/IFFT/2-D DCT pipeline architecture using the $R4^2SDF$ algorithm. The finite wordlength analysis is given in Section 6.3, and indicates that the proposed architecture achieves the required system performance in both 256-point FFT/IFFT and 8×8 2-D DCT modes with the lowest hardwire cost. Section 6.4 tabulates the comparison results in terms of hardware utilization and cost to demonstrate the high cost-efficiency of the proposed architecture, and also discusses the chip implementation. The section 6.5 draws conclusions.

6.1 8×8 2D FFT and 8×8 2D DCT Formula

Two concurrent 2D DCTs can be calculated by the single 2D shifted FFT (SFFT) algorithm [74] from the input reordering and post computation. This study presented a high-speed pipeline processor to support the triple-mode 256-point FFT/IFFT/8×8 2D DCT with the radix-4^2 algorithm. Two concurrent 2D DCTs results can be obtained by the proposed radix-4^2 based architecture in the 8×8 2D DCT mode. The 8×8 2D DCT $C[k_1, k_2]$ of the input signal $x(n_1, n_2)$ is given by

$$C[k_1,k_2]=\frac{1}{4}b(k_1)b(k_2)\sum_{n_1=0}^{7}\sum_{n_2=0}^{7}x(n_1,n_2)\cdot\cos\left(\frac{\pi(n_1+\frac{1}{2})k_1}{8}\right)\cdot\cos\left(\frac{\pi(n_2+\frac{1}{2})k_2}{8}\right). \tag{88}$$

This study neglects the post-scaling factor of $\frac{1}{4}b(k_1)b(k_2)$ in (88). The input data $x(n_1, n_2)$ could then be reordered as

$$\begin{aligned}
&y(i_1,i_2)=x(2i_1,2i_2),\\
&y(i_1,7-i_2)=x(2i_1,2i_2+1),\\
&y(7-i_1,i_2)=x(2i_1+1,2i_2),\\
&y(7-i_1,7-i_2)=x(2i_1+1,2i_2+1),
\end{aligned} \tag{89}$$

where $i_1 = i_2 = 0,1,2,3$. After the scaling and input data reordering of (89), (88) can be recast as

$$X[k_1,k_2]=\sum_{n_1=0}^{7}\sum_{n_2=0}^{7}y(n_1,n_2)\cdot\cos\left(\frac{\pi k_1(1+4n_1)}{16}\right)\cdot\cos\left(\frac{\pi k_2(1+4n_2)}{16}\right). \tag{90}$$

The value of $X[k_1, k_2]$ is then calculated with the 8×8 2D SFFT with a time-domain shift of 1/4 samples. The detail description of the transfer function between 8×8 2D SFFT and 8×8 2D DCT could be found in Appendix A.

$$Y_s[k_1,k_2]=\sum_{n_1=0}^{7}\sum_{n_2=0}^{7}y(n_1,n_2)\cdot W_8^{(n_1+\frac{1}{4})k_1}\cdot W_8^{(n_2+\frac{1}{4})k_2}$$

$$=W_8^{\frac{1}{4}k_1}\cdot W_8^{\frac{1}{4}k_2}\cdot\sum_{n_1=0}^{7}\sum_{n_2=0}^{7}y(n_1,n_2)W_8^{n_1k_1}\cdot W_8^{n_2k_2}=W_8^{\frac{1}{4}k_1}\cdot W_8^{\frac{1}{4}k_2}\cdot Y[k_1,k_2], \tag{91}$$

where $0\le k_1,k_2,n_1,n_2\le 7$. In (91), the 8×8 2D FFT $Y[k_1, k_2]$ of the input signal $y(n_1, n_2)$ is given by

$$Y[k_1,k_2]=\sum_{n_1=0}^{7}\sum_{n_2=0}^{7}y(n_1,n_2)\cdot W_8^{n_1k_1}\cdot W_8^{n_2k_2}, \tag{92}$$

where $0 \le k_1, k_2, n_1, n_2 \le 7$. Since the input data $y(n1,n2)$ form a real-valued sequence, the second half output can be derived as

$$Y_S[8-k_1,k_2] = \sum_{n_1=0}^{7}\sum_{n_2=0}^{7} y(n_1,n_2)\cdot W_8^{(n_1+\frac{1}{4})(8-k_1)} \cdot W_8^{(n_2+\frac{1}{4})k_2}$$

$$= (-j)\cdot \sum_{n_1=0}^{7}\sum_{n_2=0}^{7} y(n_1,n_2)\cdot W_8^{-(n_1+\frac{1}{4})k_1} \cdot W_8^{(n_2+\frac{1}{4})k_2} \tag{93}$$

where $0 \le k_1, k_2, n_1, n_2 \le 7$. By combining of the eqs. (91) and (93), the 8×8 2D DCT output can be recast as

$$X[k_1,k_2] = \frac{1}{2}\{\mathrm{Re}[Y_S(k_1,k_2)] - \mathrm{Im}[Y_S(8-k_1,k_2)]\} \tag{94}$$

where $0 \le k_1, k_2 \le 7$. Equation (94), adopts only the real value of $Y_S(k_1,k_2)$ and the imaginary value of $Y_S(8-k_1,k_2)$ to calculate the $X[k_1, k_2]$. By combining two reordered input sequences $\{y_1(n_1,n_2)\}$, $\{y_2(n_1,n_2)\}$ for two independent sequences $\{x_1(n_1,n_2)\},\{x_2(n_1,n_2)\}$, and forming a complex input sequence $\{y(n_1,n_2)=y_1(n_1,n_2)+jy_2(n_1,n_2)\}$, the double throughput of 2D 8×8 DCT of $\{x_1(n_1,n_2)\},\{x_2(n_1,n_2)\}$ can be derived by single 2D 8×8 SFFT computation. Consequently, two independent 8×8 2D DCTs $X_1[k_1,k_2]$, $X_2[k_1,k_2]$ of $x_1(n_1,n_2)$, $x_2(n_1,n_2)$, respectively, can then be created as

$$X_1[k_1,k_2] = \frac{1}{4}\{\mathrm{Re}[Y_s(k_1,k_2)] - \mathrm{Re}[Y_s(8-k_1,8-k_2)]\}$$
$$-\frac{1}{4}\{\mathrm{Im}[Y_s(8-k_1,k_2)] + \mathrm{Im}[Y_s(k_1,8-k_2)]\}, \tag{95a}$$

$$X_2[k_1,k_2] = \frac{1}{4}\{\mathrm{Im}[Y_s(k_1,k_2)] - \mathrm{Im}[Y_s(8-k_1,8-k_2)]\}$$
$$+\frac{1}{4}\{\mathrm{Re}[Y_s(8-k_1,k_2)] + \mathrm{Re}[Y_s(k_1,8-k_2)]\}. \tag{95b}$$

To help understand the serial pipeline operation, the 2D location $X[k_1, k_2]$ and $x(n_1, n_2)$ can be substituted as $X[8k_1+k_2]$ and $x(8n_1+n_2)$, respectively. Then, the specific two-dimensional (2D) linear index map is applied as follows:

$n_1 = 4n_{11} + n_{12}$,

$k_1 = 2k_{11} + k_{12}$,

where $0 \le n_{11} \le 1$, $0 \le n_{12} \le 3$, $0 \le n_2 \le 7$,

$0 \le k_{12} \le 1$, $0 \le k_{11} \le 3$ and $0 \le k_2 \le 7$. (96)

The word numbers of the shift registers in the post-computation of the fourth stage can be minimized by following the specific mapping in (96). The 8×8 2-D FFT CFA form can then be written as

$$Y[16k_{11}+8k_{12}+k_2]=\sum_{n_2=0}^{7}\left\{\sum_{n_{12}=0}^{3}\left\{\sum_{n_{11}=0}^{1}y(32n_{11}+8n_{12}+n_2)W_8^{4n_{11}k_{12}}\right\}W_8^{n_{12}(2k_{11}+k_{12})}\right\}W_8^{n_2k_2}$$

$$=\sum_{n_2=0}^{7}\left\{\sum_{n_{12}=0}^{3}B_{32}^{k_{12}}(8n_{12}+n_2)W_8^{n_{12}(2k_{11}+k_{12})}\right\}W_8^{n_2k_2}$$

$$=\sum_{n_2=0}^{7}B_8^{k_{12},k_{11}}(n_2)W_8^{n_2k_2}=B_{even}^{k_{12},k_{11},k_2}+W_8^{k_2}B_{odd}^{k_{12},k_{11},k_2}\,. \tag{97}$$

The butterfly structures for 8×8 2D DCT, corresponding to above equations (88)-(97), are summarized as follows:

Butterfly stage I:

$$B_{32}^{k_{12}}(8n_{12}+n_2)=y(8n_{12}+n_2)+(-1)^{k_{12}}y(8n_{12}+n_2+32)\,.$$

Butterfly stage II:

$$B_8^{k_{12},k_{11}}(n_2)=B_{32}^{k_{12}}(n_2)+W_8^{k_{12}}\left[(-j)^{k_{11}}B_{32}^{k_{12}}(n_2+8)\right]$$
$$+W_8^{2k_{12}}\left[(-1)^{k_{11}}B_{32}^{k_{12}}(n_2+16)\right]+W_8^{3k_{12}}\left[(j)^{k_{11}}B_{32}^{k_{12}}(n_2+32)\right].$$

Butterfly stage III:

$$B_{even}^{k_{12},k_{11},k_2}=\left\{B_8^{k_{12},k_{11}}(0)+(-j)^{k_2}B_8^{k_{12},k_{11}}(2)+(-1)^{k_2}B_8^{k_{12},k_{11}}(4)+(j)^{k_2}B_8^{k_{12},k_{11}}(6)\right\},$$

$$B_{odd}^{k_{12},k_{11},k_2}=\left\{B_8^{k_{12},k_{11}}(1)+(-j)^{k_2}B_8^{k_{12},k_{11}}(3)+(-1)^{k_2}B_8^{k_{12},k_{11}}(5)+(j)^{k_2}B_8^{k_{12},k_{11}}(7)\right\}.$$

The additional stage of 2-D DCT:

$$B_1^{k_{12},k_{11},k_2}(n_2)=B_{even}^{k_{12},k_{11},k_2}+W_8^{k_2}B_{odd}^{k_{12},k_{11},k_2}\,.$$

The time-domain shift stage of 2-D DCT:

$$Y_s[8k_1+k_2]=W_8^{\frac{1}{4}(k_1+k_2)}\cdot Y[8k_1+k_2]\,.$$

Butterfly stage IV:

$$X_1[8k_1+k_2]=\frac{1}{4}\{\mathrm{Re}[Y_s(8k_1+k_2)]-\mathrm{Re}[Y_s(72-8k_1-k_2)]\}$$
$$-\frac{1}{4}\{\mathrm{Im}[Y_s(64-8k_1+k_2)-\mathrm{Im}[Y_s(8+8k_1-k_2)]]\},$$

$$X_2[8k_1+k_2]=\frac{1}{4}\{\mathrm{Im}[Y_s(8k_1+k_2)]-\mathrm{Im}[Y_s(72-8k_1-k_2)]\}$$

$$+\frac{1}{4}\{\mathrm{Re}[Y_s(64-8k_1+k_2)+\mathrm{Re}[Y_s(8+8k_1-k_2)]]\}. \tag{98}$$

Two 8×8 2D DCT computation results $X_1[8k_1+k_2]$ and $X_2[8k_1+k_2]$ are calculated concurrently in the post-computation of the butterfly stage IV. The 8×8 2-D IDCT computation can also be obtained following a similar decomposition procedure. Because of the cost-effective constraint in the physical design, this study only considers the triple-mode FFT/IFFT and 2-D DCT computations. The derivation results of the radix-4^2 based FFT/IFFT/2-D DCT algorithm indicate that all butterfly computation can be easily implemented with four four-input complex adders and some shuffle circuits. The radix-4 butterfly structure has no multipliers. Additionally, the regular structure can be easily derived in both the 8×8 2-D DCT and 256-point FFT/IFFT pipeline processor architecture.

6.2 Pipeline 256-Point FFT/IFFT/8×8 2D-DCT Processor Architecture

He *et al.* presented several pipeline FFT/IFFT architectures [31]. The serial delay feedback (SDF) based architecture is known to have a low hardware cost and high cost-efficiency advantages with the feedback type shift registers architecture [31, 32, 34]. The delay-feedback type shift register approaches are always more efficient than other corresponding approaches in terms of memory utilization since the butterfly output share the same storage with its input [31]. The R2^2SDF pipeline architecture has the same computation complexity as the radix-4 algorithm, and few hardware requirements as the radix-2 algorithm. This work presents a R4^2SDF reconfigurable pipeline architecture with a low computation complexity as the radix-16 algorithm, and low hardware requirements as the radix-4 algorithm. Significantly, the proposed triple-mode radix-4 butterfly structure, like the radix-2 butterfly structure, does not require a complex multiplier or constant multiplier. Section 6.4 presents detailed comparisons between the R4^2SDF and R2^2SDF architectures. This section describes a novel radix-4^2 single-path delay feedback (R4^2SDF) architecture to support the three modes, 256-point FFT, 256-point IFFT, and 8×8 2D DCT, based on the radix-4^2 DIF FFT algorithm obtained in the previous section. Figure 31 shows a block diagram of the purposed R4^{2}SDF-based 256-point FFT/IFFT and 8×8 2-D DCT pipeline processor. Based on the proposed R4^{2}SDF pipeline architecture, the cost-effective 256-point FFT/IFFT processor is first constructed. This processor only requires four butterfly stages with 255-word shift registers, two constant multipliers and one complex multiplier with one coefficient ROM, represented by black solid color in Fig. 31. Figure 31 indicates that a single data stream passes through two constant multipliers and one complex multiplier to realize different combinations of k_1, k_2 and k_3 of $X[k_1+4k_2+16k_3]$, as illustrated in (90). Using some control circuits, one additional radix-2 butterfly (R2-BF) with an one-word shift register and additional eight-word shift register, two concurrent 2D-DCT operations are calculated from the single 2D-SFFT computation as depicted in (89), (95a), and (95b). These extra circuits were embedded at the first butterfly stage, the additional stage and the fourth butterfly stage, which is represented by the gray color in Fig. 31. These circuits complete the input reordering, time domain shift and post-computation in the 8×8 2-D DCT computation, respectively. Notably, only one radix-2 butterfly and nine-word shift register are needed to support additional two 8×8 2D DCT computations in the original pipeline SDF-based FFT/IFFT architecture. The proposed architecture has, in total, four radix-4 butterflies, one radix-2 butterfly, two constant multipliers, one complex multiplier, one coefficient ROM and a 264-word shift register. To help understand the corresponding functions of each building block, the respective equation numbers related to each element are shown in Table 11. The detailed operations of each element are described as follows.

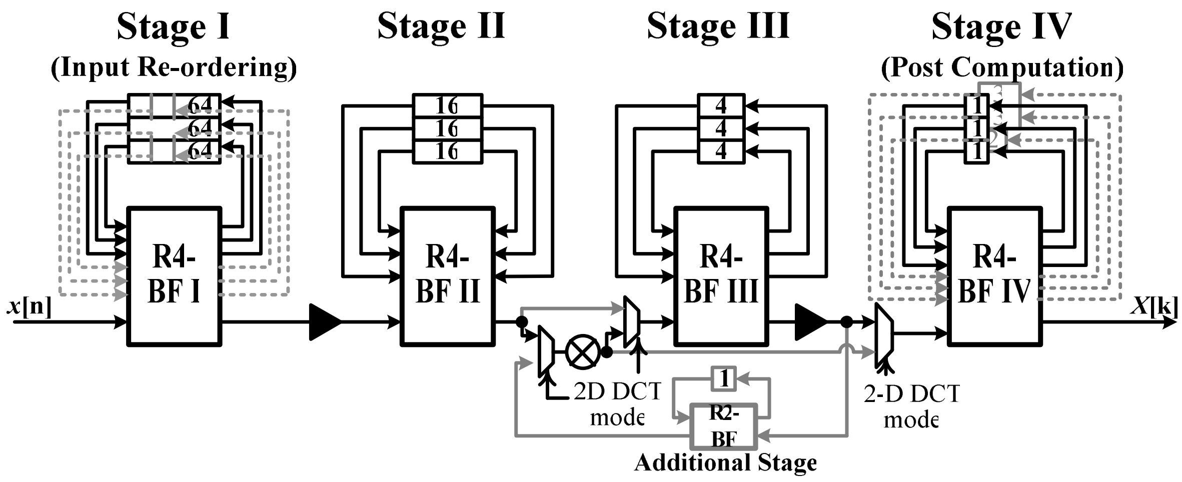


Fig. 31: Block diagram of the R4^{2}SDF-based 256-point FFT/IFFT and 8×8 2D-DCT architecture.

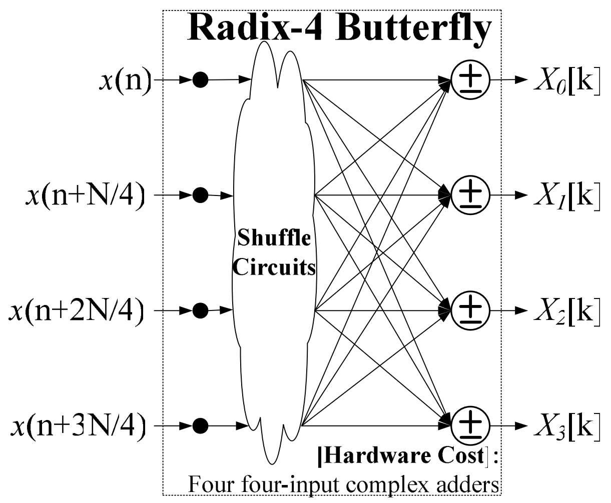


Fig. 32: Block diagram of the radix-4 butterfly architecture.

Table 11 The Corresponding Equation Numbers for Each Building Block.

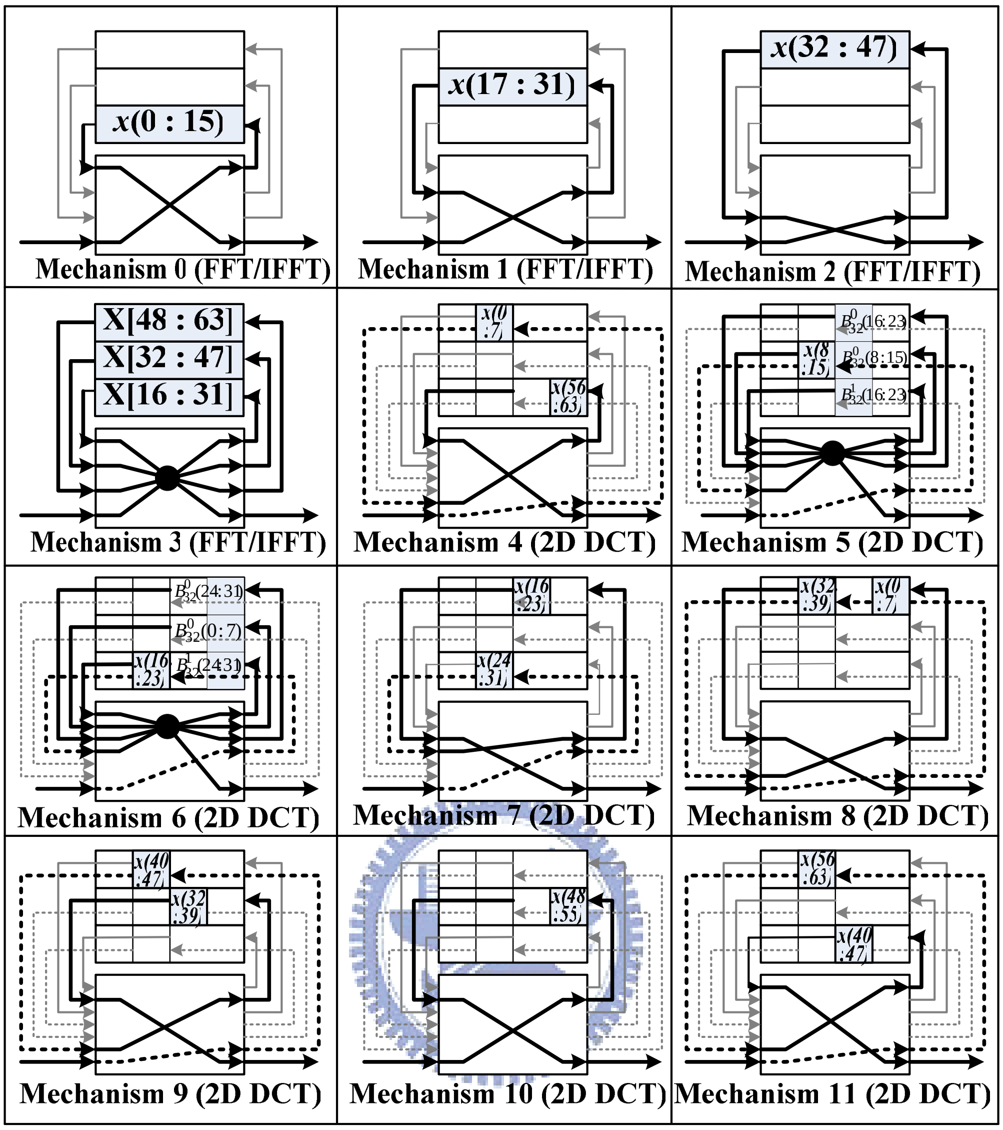
Building Blocks	BFI	Cons. Mult. I	BFII	Comp. Mult.	BFIII	Cons. Mult. II	BFIV	R2-BF
FFT Eq. #	(79)	(81)	(81)	(80)	(79)	(81)	(81)	N/A
IFFT Eq. #	(84a)	(84b)	(84b)	(83)	(84a)	(84b)	(84b)	N/A
DCT Eq. #	(89), (97)	(97)	(97)	(91)	(97)	(97)	(95)	(97)

6.2.1 Radix-4 Butterfly and Radix-2 Butterfly

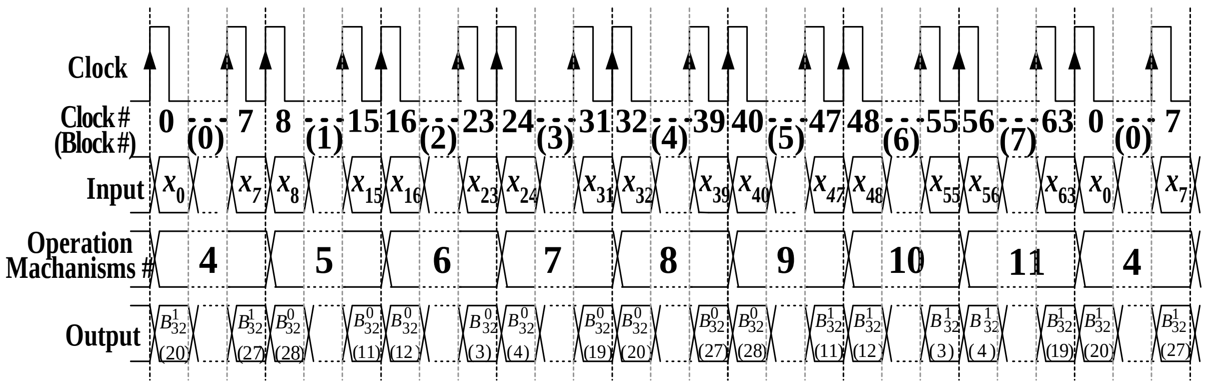
The derivation results of the radix-4^2 based algorithm reveal that both the FFT/IFFT butterfly computation in (79), (81), (84a), (84b), and the 8×8 2D-DCT butterfly computation in (98), can be easily completed with the radix-4 based butterfly architecture. The only difference between the 8×8 2D-DCT and the 256-point FFT/IFFT butterfly computation is the summands and minuends at the butterfly stage one and four, which can be easily realized with the multiplex circuits in the radix-4 butterfly structure. Significantly, the number of the two first stages in the 8×8 2D-DCT computation can be completed concurrently at the first butterfly stage in parallel. Additionally, the input reordering operation at the first stage and the post-computation operation at the fourth stage of the 8×8 2D-DCT mode are described in detail. Figure 32 illustrates the proposed radix-4 butterfly structure, which only includes four four-input complex adders with no complex multipliers inside. This configuration means that the proposed radix-4 butterfly structure has a low hardware cost of higher-radix butterfly structures. Moreover, the proposed radix-4^2 algorithm has the same complex multiplication complexity as the radix-16 algorithm, so has the high cost efficiency of lower radix architectures. To obtain the additional stage of the 8×8 2D DCT mode in (98), one additional SDF-based radix-2 butterfly structure with one word shift register is required, as illustrated in Fig. 31. The additional stage of the 2D DCT mode only requires two 2-input complex adders and one shift register, giving it a small hardware penalty.

6.2.2 Memory Structure

The memory structure of butterfly stage is well known to be an important issue for the high cost-effective FFT/IFFT pipeline processor design. From the exiting researches, there are mainly two different approaches: delay commutator (DC) [31] and delay feedback (DF) [31, 32, 34]. In this study, the DF based memory structure is adopted and depicted in Fig. 31. In order to compute the radix-4 based butterfly computations, the input data and the intermediate results have to be reordered as four concurrently data streams using memory as shown in Fig. 32. Each radix-4 butterfly unit applies the three parallel memories to store the serial data input and butterfly output in the feedback paths as presented in Fig. 31. The timing sequence of N-point FFT computation can be divided into four stages, each containing N/4 clock cycles. In the first N/4 cycles (i.e. first stage), the butterfly units simply store the input samples into the first feedback memory. Similarity, the second and third feedback memory are filled in the second and third stages. After the 3N/4 cycles, the butterfly units retrieves the $x(n)$, $x(n+\mathrm{N}/4)$ and $x(n+2\mathrm{N}/4)$ samples from the feedback memory, performs corresponding operations with the sample $x(n+3\mathrm{N}/4)$ and then feeds the output into the next butterfly units as depicted in Fig. 31. The required number of memory cells for the kth stage is $3\times\mathrm{N}/(4^k)$. Thus, the 256-points FFT/IFFT computations require the 64×3, 16×3, 4×3 and 1×3 word shift registers in the first, second, third and fourth butterfly stages, respectively. Significantly, the SDF based pipeline FFT/IFFT structure is highly regular, which has the high effective memory structure with the simpler routing complexity [31, 32, 34]. In this study, the shift registers were all realized by the cascaded flip-flops, which are composed of two latch circuits.

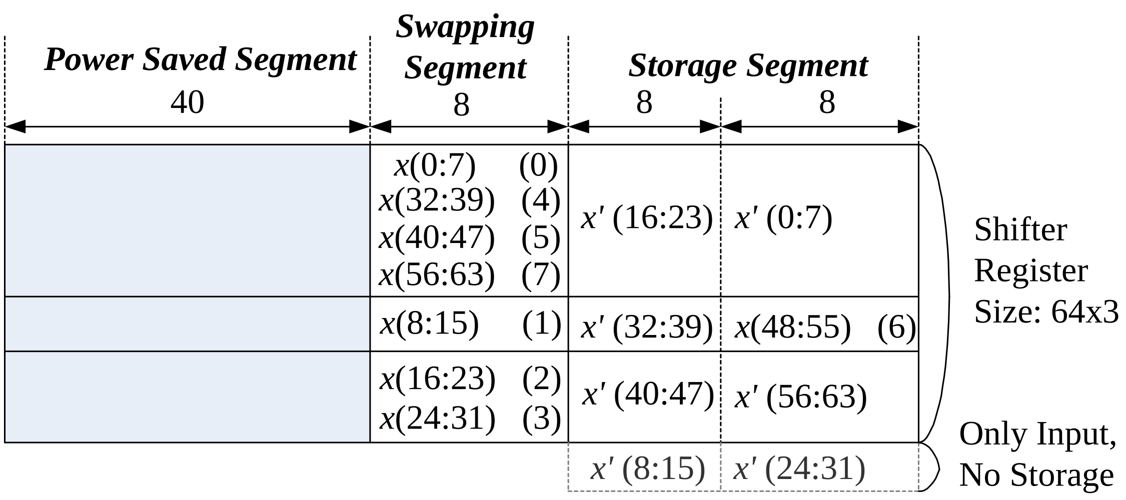


(a) The proposed 12 reconfigurable operation mechanisms of the first butterfly stage.



Clock
Clock# (Block#)
0 (0) 7 8 (1) 15 16 (2) 23 24 (3) 31 32 (4) 39 40 (5) 47 48 (6) 55 56 (7) 63 0 (0) 7
Input
x0 x7 x8 x15 x16 x23 x24 x31 x32 x39 x40 x47 x48 x55 x56 x63 x0 x7
Operation Machanisms #
4 5 6 7 8 9 10 11 4
Output
B132(20) B132(27) B032(28) B032(11) B032(12) B032(3) B032(4) B032(19) B032(20) B032(27) B032(28) B132(11) B132(12) B132(3) B132(4) B132(19) B132(20) B132(27)

(b) The timing sequences of operation mechanism in the first butterfly stage.



(c) The storage content in SSR in the 8×8 2D-DCT mode.

$B_{32}^{0}(16)$	$B_{32}^{0}(23)$	$B_{32}^{0}(17)$	$B_{32}^{0}(22)$	$B_{32}^{0}(18)$	$B_{32}^{0}(21)$	$B_{32}^{0}(19)$	$B_{32}^{0}(20)$	$B_{32}^{0}(24)$	$B_{32}^{0}(31)$	$B_{32}^{0}(25)$	$B_{32}^{0}(30)$	$B_{32}^{0}(26)$	$B_{32}^{0}(29)$	$B_{32}^{0}(27)$	$B_{32}^{0}(28)$
$B_{32}^{1}(8)$	$B_{32}^{1}(15)$	$B_{32}^{1}(9)$	$B_{32}^{1}(14)$	$B_{32}^{1}(10)$	$B_{32}^{1}(13)$	$B_{32}^{1}(11)$	$B_{32}^{1}(12)$	$B_{32}^{1}(0)$	$B_{32}^{1}(7)$	$B_{32}^{1}(1)$	$B_{32}^{1}(6)$	$B_{32}^{1}(2)$	$B_{32}^{1}(5)$	$B_{32}^{1}(3)$	$B_{32}^{1}(4)$
$B_{32}^{1}(16)$	$B_{32}^{1}(23)$	$B_{32}^{1}(17)$	$B_{32}^{1}(22)$	$B_{32}^{1}(18)$	$B_{32}^{1}(21)$	$B_{32}^{1}(19)$	$B_{32}^{1}(20)$	$B_{32}^{1}(24)$	$B_{32}^{1}(31)$	$B_{32}^{1}(25)$	$B_{32}^{1}(30)$	$B_{32}^{1}(26)$	$B_{32}^{1}(29)$	$B_{32}^{1}(27)$	$B_{32}^{1}(28)$
$B_{32}^{0}(8)$	$B_{32}^{0}(15)$	$B_{32}^{0}(9)$	$B_{32}^{0}(14)$	$B_{32}^{0}(10)$	$B_{32}^{0}(13)$	$B_{32}^{0}(11)$	$B_{32}^{0}(12)$	$B_{32}^{0}(0)$	$B_{32}^{0}(7)$	$B_{32}^{0}(1)$	$B_{32}^{0}(6)$	$B_{32}^{0}(2)$	$B_{32}^{0}(5)$	$B_{32}^{0}(3)$	$B_{32}^{0}(4)$

(d) The content of the 8×8 2-D DCT computation result in SSR.

Fig. 33: Block diagram of the proposed first radix-4 butterfly stage in the R4^{2}SDF-based 256-point FFT/IFFT and 8×8 2D-DCT architecture.

6.2.3 Input Re-ordering and First Butterfly Computation

Consider the new radix-4^2 algorithm presented in section 6.1. The proposed SDF architecture is estimated to need 64×3-word shift registers at the first butterfly stage in the 256-point FFT/IFFT mode. Although the 8×8 2D DCT mode only requires 16×3-word shift registers at the first butterfly stage, the 8×8 2D DCT mode needs a swapping buffer to complete the input re-ordering and post-computation in (89) and (95) from the 8×8 2D SFFT computation. Notably, the number of shift registers at the fourth butterfly stage for the post computation in the 8×8 2D DCT mode depends on the sequential order of the input data at the first butterfly stage. Following the specific linear mapping in (96), the number of shift registers can be reduced to only eight words at the fourth butterfly stage, as revealed in Fig. 31. Comparing with the other linear mapping, the proposed architecture could reduce at least 96% shift registers cost. The segmented shift registers (SSR) structure is also proposed to realize both the input re-ordering and butterfly computation operation at the first stage to support the 256-point FFT/IFFT and 8×8 2D DCT modes.

Figure 33(a) shows the 12 proposed operation mechanisms of the first butterfly stage to finish the input reordering and the first-stage computation. Operation mechanisms 0–3 are adopted in the FFT/IFFT computation, and operation mechanisms 4–11 are adopted in the 8×8 2D DCT computation. In the 8×8 2D DCT mode, reconfigurable operation mechanisms 5 and 6 are adopted for the butterfly computation, and reconfigurable operation mechanisms 4 and 7–11 are adopted for the input reordering. Figure 33(b) lists the corresponding timing sequence of the first butterfly stage, which discusses the relationships among the input data, output data and respective operation mechanisms during each clock (block number) in the 8×8 2D DCT mode. Additionally, Figs. 33(c) and 33(d) illustrate the data content before and after the butterfly computation in SSR, respectively. The first stage butterfly computation is completed by applying operation mechanisms 5 and 6. Most results of 8×8 2D DCT computation are then pushed back into the SSR, as shown in Fig. 33(d), where $B_{32}^{k_{12}}$ denotes the computation results from (98). Fig. 33(d) presents the complete computation results. The 64×3-word shift register is segmented as (40+8+16)×3, which is easily realized by three dependent clock domains with a simple 3-bit counter controller, as depicted in Fig. 33(c). These three segments in the SSR are called the power-saving, swapping and storage segments, and their sizes are 40×3, 8×3 and 16×3, respectively. Since 2D DCT mode as depicted in (97) has a low computation complexity, the first, second and third butterfly stages have shift registers comprising 40×3, 8×3 and 2×3 words, respectively. These shift registers are set as power-saving segments, and gated to reduce power consumption. To perform the input re-ordering operation, 64 serial input data words are split into eight blocks of eight words, as shown in Figs. 33(b) and 33(c). The block numbers, written inside the brackets in Figs. 33(b) and 33(c), denote the serial input sequential order of eight-word blocks. In Fig. 33(c), the terms x' and x respectively represent the 2D DCT image data in the previous and current frame, which both contain 64 points in each frame. Following the operation mechanisms 4, 7, 8, 9, 10 and 11 in Fig. 33(a), the serial input data of each block adopt the swapping segment as the swapping space to achieve the required storage ordering in the storage segment.

The detail timing sequence of the proposed 8×8 2D DCT computations is given as follow. Operation mechanism 4 pushes the input data $x(0{:}7)$ into the swapping segment from the clock number 0 to 7 (block number 0). At the same time, the original data $x'(56{:}63)$ in the swapping segment are simultaneously pushed into the storage segment as illustrated in Figs. 33(a) and 33(c). In the following 16 clock cycles, operation mechanisms 5 and 6 replace the swapping segments with the input data $x(8{:}15)$ and $x(16{:}23)$ (i.e. block number 1 and 2), as presented in Figs. 33(a), 33(b) and 33(c). Operation mechanisms 5 and 6 provide the original swapping segment data $x'(8{:}15)$ and $x'(24{:}31)$ for the first butterfly stage computation in (98), along with the data in the storage segment. At the same time, 48 new 8×8 2D DCT results of $B_{32}^{k_{12}}(8n_{12}+n_2)$, as listed in the top 3 rows of Fig. 33(d), are pushed into the storage segment by the feedback path. Furthermore, the other 16 new 8×8 2D DCT results, which are listed in the final row in Fig. 33(d), are pushed directly to the second butterfly stage, as shown in Fig. 33(b). Notably, the 48 different 2D DCT results in the storage segment are pushed out one-by-one due to the swapping operation by the swapping segment data in the following 48 clock cycles. Following a similar procedure, the serial input data of block numbers 3, 4, 5 and 7 complete their respective swapping operations by operation mechanisms 7, 8, 9 and 11. The block number 6 is stored into the storage segment directly by operation mechanism 10 as illustrated in Figs. 33(a) and 33(c). The input re-ordering operation is finished after a period of 64 clock cycles, which includes 7 clock cycles of input swapping latency.

6.2.4 Constant Multiplier

Based on the derivation results in Section II, the radix-4^2 algorithm requires some complex multiplications, namely $W_{16}^{k_1}(W_{16}^{-n_1}), W_{16}^{2k_1}(W_{16}^{-2n_1}), W_{16}^{3k_1}(W_{16}^{-3n_1})$ in the 256-point FFT/IFFT mode in (81), (84), and $W_8^{k_{12}}, W_8^{2k_{12}}, W_8^{3k_{12}}, W_8^{k_2}$ in the 8×8 2D DCT mode in (98). Due to the finite range of k_1 and n_1 in Eqs. (81) and (84b), namely 0–3, the three complex multiplications, $W_{16}^{k_1}$ ($W_{16}^{-n_1}$), $W_{16}^{2k_1}$ ($W_{16}^{-2n_1}$) and $W_{16}^{3k_1}$ ($W_{16}^{-3n_1}$) can be written as $\{W_{16}^{0}(W_{16}^{-0}), W_{16}^{1}(W_{16}^{-1}), W_{16}^{2}(W_{16}^{-2}), W_{16}^{3}(W_{16}^{-3})\}$, $\{W_{16}^{0}(W_{16}^{-0}), W_{16}^{2}(W_{16}^{-2}), W_{16}^{4}(W_{16}^{-4}), W_{16}^{6}(W_{16}^{-6})\}$ and $\{W_{16}^{0}(W_{16}^{-0}), W_{16}^{3}(W_{16}^{-3}), W_{16}^{6}(W_{16}^{-6}), W_{16}^{9}(W_{16}^{-9})\}$. Following the similar procedure, $W_8^{k_{12}}$, $W_8^{2k_{12}}$, $W_8^{3k_{12}}$ and $W_8^{k_2}$ in (98) can be expanded as $\{W_8^0, W_8^1\}$, $\{W_8^0, W_8^2\}$, $\{W_8^0, W_8^3\}$ and $\{W_8^0, W_8^1, W_8^2, W_8^3, W_8^4, W_8^5, W_8^6, W_8^7\}$. The system has in total 38 different twiddle factor values, which could be implemented as 38 different constant multipliers by only shifters and adders. Based on the SDF based architecture, the proposed design only has to calculate one complex multiplication in Eqs. (81), (84) and (97) during each clock cycle. The 38 twiddle factor values can thus be reduced to the extension of two different values of W_{16}^1 and W_{16}^2 using the complex conjugate symmetry rule. Accordingly, the other 36 twiddle factor values can be expressed as the real-imaginary swapping or sign inversion of these two constant values. Moreover, the repeated shifters and adders of two constant multipliers could be simplified using the subexpression elimination algorithm [65] as illustrated in Fig. 34. According to our implementation results, the small cost penalty for the multiplexer control (i.e. S0, S1 and S2) could be neglected as shown in Fig. 34.

Following the three steps to reduce the complex multipliers to the most economical constant multipliers are summarized as below. First, the twiddle factors from Eqs. (81), (84) and (98) are realized as the constant multipliers, which only contain shifters and adders as shown in Fig. 31. Second, the complex conjugate symmetry rule is applied to decrease the number of complex multiplications (90) to only two constant multiplications per stage with some shuffle circuits as shown in Fig.5, thus achieving a constant multiplier cost reduction of 94.7%. Finally, the subexpression elimination algorithm [65] is adopted to reduce the number of shift circuits by more than 20%, and the number of complex adders by 50% in the one constant multiplier, as depicted in Fig. 34. The strictest constant multiplier is obtained in the purposed architecture by following these three steps. The cost penalty of the constant multiplier is thus minimized.

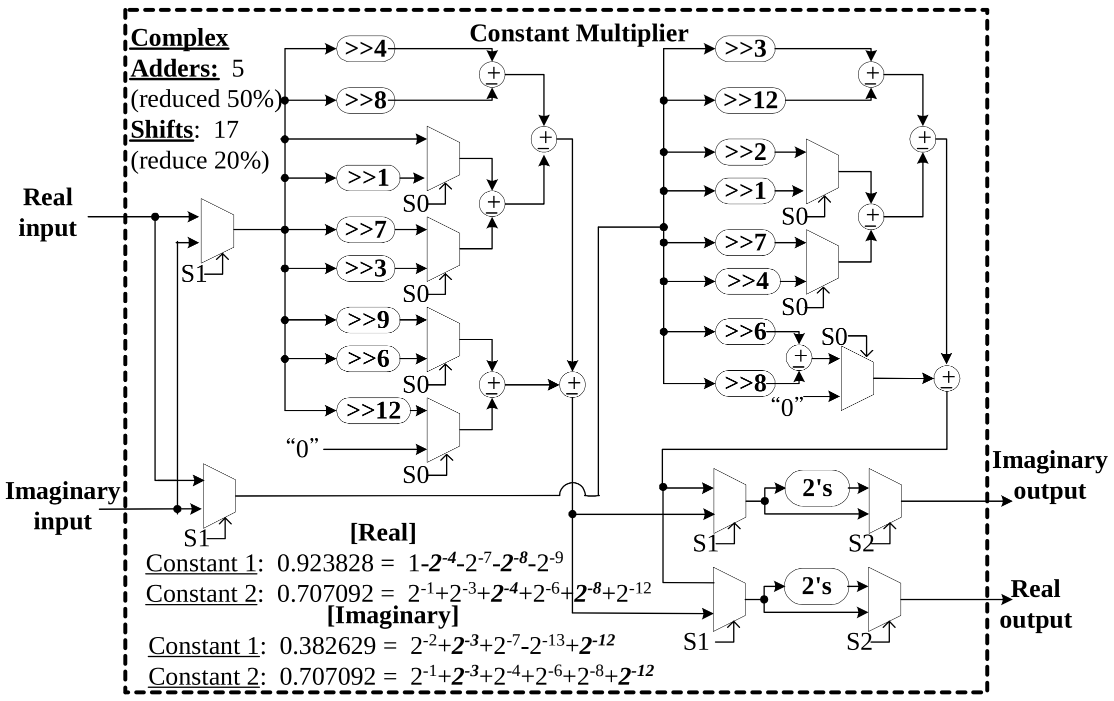


Fig. 34: Block diagram of the proposed constant multiplier architecture.

6.2.5 Eight-Folded Complex Multiplier

The proposed architecture has only one complex multiplier and one coefficient ROM to realize the complex multiplication of twiddle factors $W_N^{n_3(k_1+4k_2)}$ in (78), $W_N^{-k_3(n_1+4n_2)}$ in (83) and $W_8^{\frac{1}{4}(k_1+k_2)}$ in (98). Significantly, the implementation of the time-domain shift for 8×8 2D-DCT computation needs one feedback path. To decrease the ROM size, the complex conjugate symmetry rule and subexpression elimination [65] is applied to devise one eight-folded complex multiplier as shown in Fig. 35. The proposed eight-folded complex multiplier only has to store 32 words in the coefficient ROM, reducing the ROM size by 87.5%. The ROM address and data control circuit are also easily realized by the 8-bit counter controller given in Table 12.

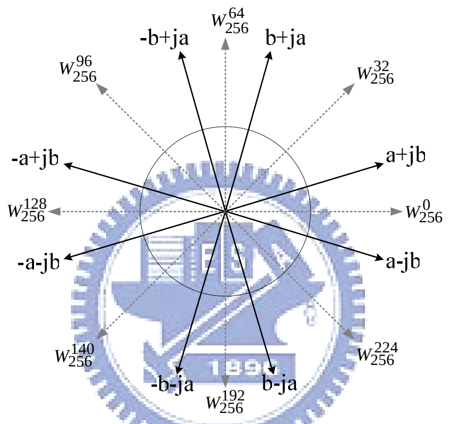


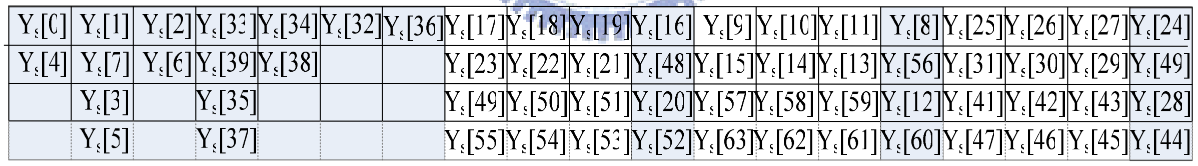
Fig. 35: The block diagram of eight-folded algorithm in the coefficient ROM.

Table 12 The Data Control of The Coefficient ROM.

$H = n_3(k_1+4k_2)$	Address Mode (H[5])	ROM address	Data Mode (H[7:5])	ROM data
0~32	0	Two's complement of H[5:0]	0	a+jb
33~63	1	H[5:0]	1	b+ja
64~95	0	Two's complement of H[5:0]	2	-b+ja
96~127	1	H[5:0]	3	-a+jb
128~159	0	Two's complement of H[5:0]	4	-a-jb
160~191	1	H[5:0]	5	-b-ja
192~223	0	Two's complement of H[5:0]	6	b-ja
224~255	1	H[5:0]	7	a-jb

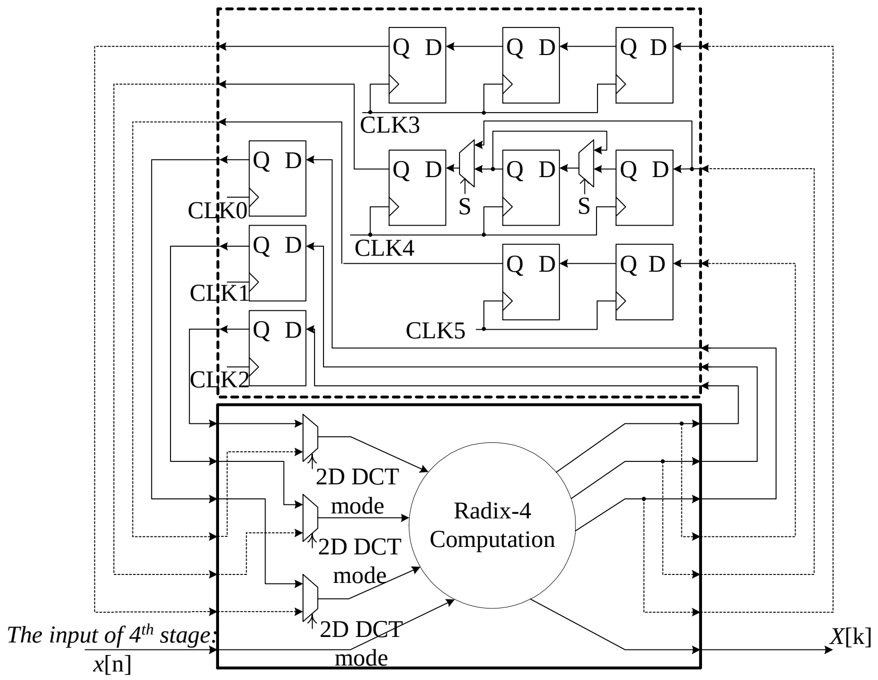
6.2.6 Post Computation

Clearly, the 256-point FFT/IFFT modes only require 1×3 word shift registers at the fourth butterfly stage of the proposed R4^{2}SDF architecture. However, the 8×8 2D DCT mode has to implement the post-computation at the fourth butterfly stage in (95a) and (95b). As described in Subsection 6.2.1, the proposed architecture follows the specific linear mapping in (97) to minimize the number of shift registers at the fourth stage. Figure 36(a) depicts the analysis of the order of the fourth butterfly results following the specific linear mapping. Notably, the gray solid line in Fig. 36(a) represents the input data order that do not follow the required sequence. For instance, $\{Y_S[17],\ Y_S[23]\}$, $\{Y_S[18],\ Y_S[22]\}$ and $\{Y_S[19],\ Y_S[21]\}$ should be regarded as three groups for the fourth butterfly computation. However, the sequence of the input data at the fourth butterfly stage is $Y_S[17]$, $Y_S[18]$, $Y_S[19]$, $Y_S[21]$, $Y_S[22]$, $Y_S[23]$. Then, $Y_S[23]$ and $Y_S[21]$ should be re-ordered. Thus, the proposed overturn shift register (OSR) structure at fourth butterfly stage resolves this simple re-ordering procedure without any performance degradation, as depicted in Fig. 36(b). The desired ordering is obtained with the OSR structure at the fourth butterfly stage, along with the input re-ordering operation at the first butterfly stage as discussed in Subsection 6.2.1. The full-pipeline R4^{2}SDF architecture can then easily follow the two concurrent 8×8 2D DCT outputs.



$Y_s[0]$	$Y_s[1]$	$Y_s[2]$	$Y_s[33]$	$Y_s[34]$	$Y_s[32]$	$Y_s[36]$	$Y_s[17]$	$Y_s[18]$	$Y_s[19]$	$Y_s[16]$	$Y_s[9]$	$Y_s[10]$	$Y_s[11]$	$Y_s[8]$	$Y_s[25]$	$Y_s[26]$	$Y_s[27]$	$Y_s[24]$
$Y_s[4]$	$Y_s[7]$	$Y_s[6]$	$Y_s[39]$	$Y_s[38]$			$Y_s[23]$	$Y_s[22]$	$Y_s[21]$	$Y_s[48]$	$Y_s[15]$	$Y_s[14]$	$Y_s[13]$	$Y_s[56]$	$Y_s[31]$	$Y_s[30]$	$Y_s[29]$	$Y_s[49]$
	$Y_s[3]$		$Y_s[35]$				$Y_s[49]$	$Y_s[50]$	$Y_s[51]$	$Y_s[20]$	$Y_s[57]$	$Y_s[58]$	$Y_s[59]$	$Y_s[12]$	$Y_s[41]$	$Y_s[42]$	$Y_s[43]$	$Y_s[28]$
	$Y_s[5]$		$Y_s[37]$				$Y_s[55]$	$Y_s[54]$	$Y_s[53]$	$Y_s[52]$	$Y_s[63]$	$Y_s[62]$	$Y_s[61]$	$Y_s[60]$	$Y_s[47]$	$Y_s[46]$	$Y_s[45]$	$Y_s[44]$

(a) The data context of the fourth butterfly stage in the 8×8 2D DCT mode.



(b) The OSR structure of the fourth butterfly stage.

Fig. 36: Block diagram of the proposed fourth butterfly stage in the R4^{2}SDF-based 256-point FFT/IFFT and 8×8 2D-DCT architecture.

6.3 Finite Wordlength Analysis

The next generation mobile-multimedia system can handle high-quality multimedia operations with embedded 256-point FFT/IFFT and 8×8 2D DCT pipeline processor [3]-[5]. The system performance should then satisfy the relative specifications. A higher system performance undoubtedly implies a larger chip cost and greater power consumption, owing to the wider internal wordlength. Since the chip cost and system performance are known to be a trade-off, this study performed a finite wordlength analysis to estimate the appropriate word-length for both 256-point FFT/IFFT and 8×8 2D DCT system requirements.

6.3.1 Pipeline 256-Point FFT/IFFT

In the 256-point FFT/IFFT modes, the output signal to noise ratio (SNR) performance was estimated under different noise environment. First, the input data of the double floating-point precision were generated from the ideal IFFT(FFT) model by passing the additive white Gaussian noise (AWGN) channel model under five noise levels: 20dB, 40dB, 60dB, 80dB and 100dB. The input data with noise were sent into the proposed R4^{2}SDF pipeline FFT/IFFT architecture, which was modeled at different fixed-point levels. The output SNR was obtained by comparing the original input data with the fixed-point model output. The results after 100,000 iterations were averaged as depicted in Fig. 37, where the *x*-axis and *y*-axis represent the data word-length and the whole system output SNR, respectively. These analytical results demonstrate that the output SNR saturated as the data word length increased. The output SNR was increased by 20dB for each additional three bits. The 13-bit internal wordlength for each function units produced satisfactory results under 40dB noise environments, satisfying the IEEE 802.16e WiMAX [44] standard.

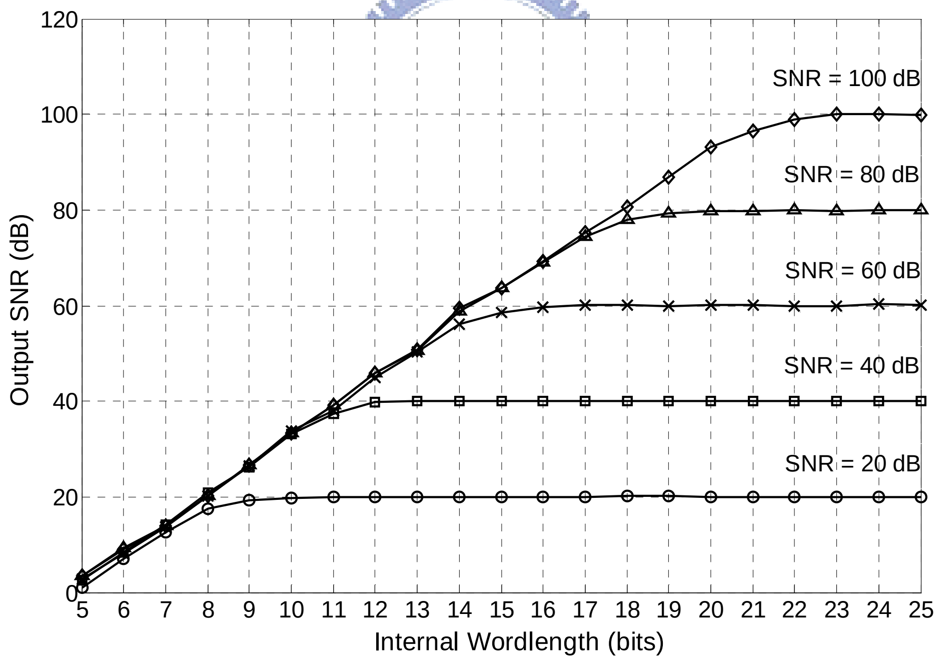


Fig. 37: Finite wordlength analysis of the proposed pipeline R4^{2}SDF-based 256 points FFT/IFFT architecture.

6.3.2 Pipeline 8×8 2-D DCT

In the 8×8 2-D DCT mode, the performance of the proposed R4^{2}SDF pipeline architecture was measured in common video compression standards, including the high-quality DV standard [75]. The DV standard defines some tolerances that the 8×8 2-D DCT computation maintains the accuracy and consequently an acceptable reconstructed video quality [75][76]. The DV standard applies four measured error criteria, namely the probability of occurrence of error, mean square errors (MSE), peak mean square error (PMSE) and steady AC coefficients [76]. Following the procedure in the preceding subsection, the double floating-point precision is assumed to be precise in comparing with the fixed-point computation. The zero-mean white input sequences are generated by a random-number generator in the range [−128, 127]. After the repeated 100,000 loops, the probability of occurrence of error, which is greater than 1, is less than 1×10^{-15}. Moreover, the steady AC coefficients of the proposed fixed-point 2D 8×8 DCT model are all zero under the equal-values input. Figures 38(a) and 38(b) depict the MSE and PMSE simulation results, respectively. Notably, the proposed architecture could satisfy the limitation of MSE and PMSE of the DV standard, when the internal wordlength is greater than 12 bits. Thus, the 13-bit internal word length for each function units is the qualified internal wordlength for the DV standard. Figures 38(c) and 38(d) indicate that the overall mean error (OME) is below 0.01, and the peak signal to noise ratio (PSNR) is close to 60dB, which has the required video compression quality under the configuration of the 13-bit internal wordlength [77]. According to the finite wordlength analysis of the proposed R4^{2}SDF 256-point FFT/IFFT pipeline architecture a 13-bit internal wordlength achieves the satisfactory results under the 40dB noise quality, thus satisfying the IEEE 802.16e standard. The 13-bit internal wordlength was thus chosen for the proposed R4^{2}SDF 256-point FFT/IFFT/2-D DCT RSoC IP to meet the requirements of next-generation handheld applications.

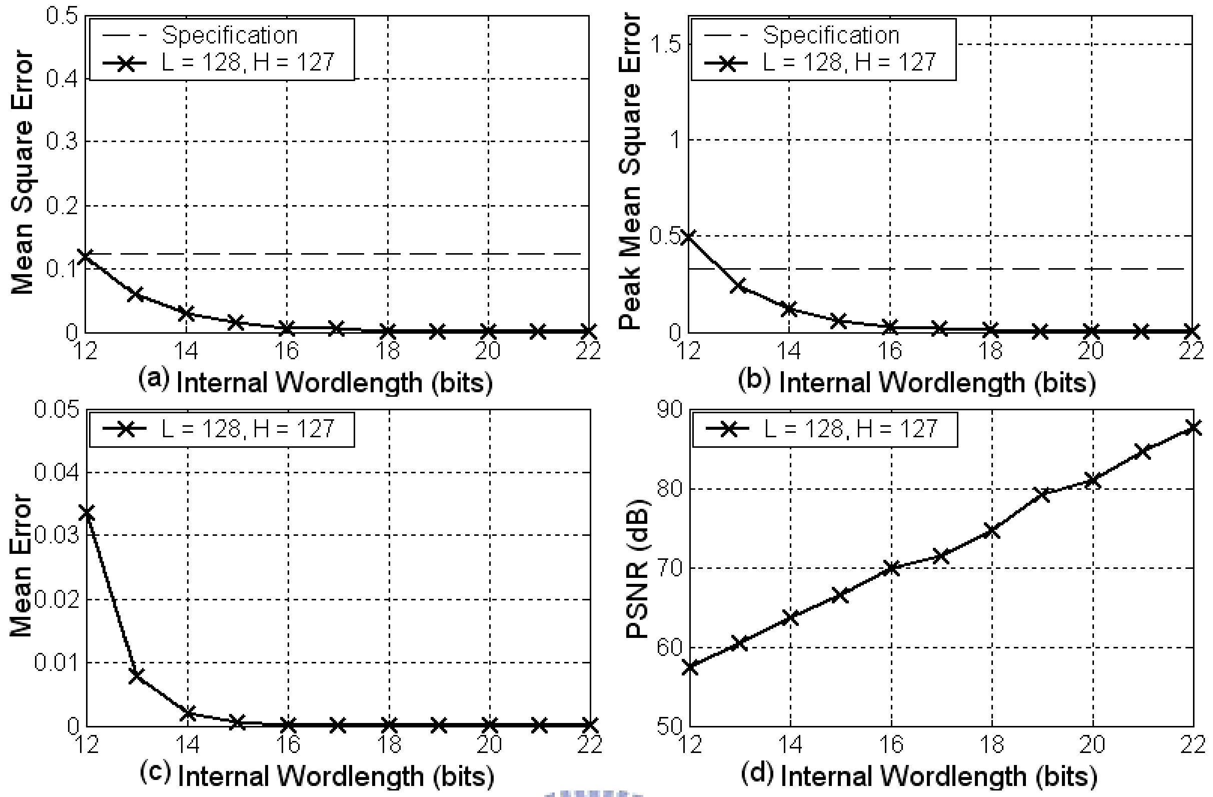


Fig. 38: Finite wordlength analysis of the proposed pipeline R4^2SDF-based 8×8 2D DCT architecture. (a). Overall mean square error analysis. (b) Peak Mean Square Error analysis. (c). Overall Mean Error analysis. (d). Peak Mean Error analysis.

6.4 Comparison and Chip Implementation

6.4.1 Comparison between R4^{2}SDF and R2^{2}SDF

He *et al.* presented the efficient pipeline FFT processor, several reliable architectures and the detailed comparison of their hardware costs [31]. A comparison of these architectures indicates that R2^{2}SDF has the highest butterfly utilization of 50%, a the highest complex multiplier utilization of 75%, and the lowest hardware resource requirement [31][34]. Additionally, the SDF-based design has the structural merits of high regularity and modularity with simple wiring complexity, making it very appropriate for the VLSI implementation of the pipeline FFT processor design [31, 32, 34]. This section presents the comprehensive comparison results of several famous pipeline FFT/IFFT architectures to demonstrate the high cost-efficiency of the proposed R4^{2}SDF FFT/IFFT architecture. The architectures were compared in two indices, namely cost and utilization, to express the cost efficient of the proposed FFT/IFFT architecture, as listed in Tables 13 and 14. Table 13 lists the required hardware resources, where T denotes the number of complex adders required in the implementation of the constant multiplier. Significantly, the proposed constant multiplier is minimized using complex conjugate symmetry rule and subexpression elimination algorithm. The area of the complex multiplier is known to be one dominant cost index in the pipeline FFT/IFFT design. The comparison results in Table 14 clearly demonstrate that the proposed R4^{2}SDF based-FFT/IFFT architecture has the fewest complex multipliers requirement among other pipeline architectures. The 256-point FFT/IFFT architecture only needs one complex multiplier, which is 67% and 95% below the requirement of the R2^{2}SDF and R8MDC FFT/IFFT architectures, respectively. Additionally, the proposed architecture applies the feedback type memory structure to maintain the minimum shift registers requirement. Although the proposed R4^{2}SDF based architecture needs slightly more complex adders than the R2^{2}SDF based architecture, this small cost penalty is acceptable.

To estimate the total chip cost in the 256-point FFT/IFFT architectures, which includes the number of complex multipliers, complex adders and memory size, the conventional comparative methodology [26, 32] with the unit of equivalent adders was adopted to estimate the cost of each different architecture. Based on the implementation results in our process, we convert the area of each complex multiplier and complex memory to the 50 and 1.3 complex adder, respectively, when adopting 13-bit precision, and the scheme with three real multiplications and five real additions, in the implementation. The rightmost column of Table 13 lists the area indexes of the equivalent adder of the 256-point FFT/IFFT architecture. Clearly, the proposed R4^2SDF-based 256-point FFT/IFFT architecture has the lowest hardware requirements. The R4^2SDF-based 256-point FFT/IFFT architecture has a 16% lower cost than the R2^2SDF-based 256-point FFT/IFFT architecture. Significantly, the cost advantage of our proposed architecture becomes more evident when the transform length is larger. Thus, the proposed R4^2SDF-based architecture has a lower hardware cost than R2^2SDF and other famous pipeline FFT/IFFT architecture in terms of the number of ROMs, complex multipliers, complex adders, constant multipliers and shift registers.

Table 14 shows the comprehensive comparison of the hardware utilization rate in terms of the utilization rate of complex multipliers, complex adders and complex memory. Clearly, the proposed architecture achieves the highest complex multiplier utilization rate among pipeline architecture (87.5%). Additionally, the proposed architecture maintains the maximum complex memory utilization rate of 100%. Furthermore, the proposed architecture, including the constant multipliers, has the highest complex adder utilization rate of 56.25%. Thus, the purposed architecture achieves a higher hardware utilization rate than R2^2SDF and other well-known pipeline FFT/IFFT architecture in terms of the utilization rate of complex multipliers, complex adders, constant multipliers and complex memory. Although the R2MDC, R4MDC and R8MDC architectures have the higher throughput rate (output/cycle) of 2, 4 and 8 than SDF based architecture, these approaches require large hardware requirement, such as complex multipliers, adders and memory size, as shown in Table 13. Therefore, this investigation focuses on the "hardware-oriented" architecture, in which the arithmetic operations can be tightly scheduled for efficient hardware utilization. This study demonstrates that the proposed R4^2SDF based pipeline FFT/IFFT architecture has the lowest hardware cost and highest hardware utilization. Conversely, the proposed R4^2SDF based pipeline FFT/IFFT architecture is the most cost-efficient.

6.4.2 8×8 2-D DCT Comparison

Many DCT implementations exist spanning a broad spectrum of architectures, focusing on different applications. Lee *et al.* [78] presented a highly parallel approach with high arithmetic cost and high power consumption for the high-performance application. The systolic implementation of Lee *et al.* [78] employs the row-column decomposition to derive the configurable 2D $N \times N$ DCT in three steps with each step implemented in systolic form. This work concentrates on high-speed FFT/IFFT/2D DCT architectures with a throughput rate of at least one output sample per cycle, targeted for applications in next-generation handheld devices needing a high data-processing rate. Moreover, the proposed architecture has high cost efficiency and low cost in a portable consumer device. This subsection lists the hardware requirement comparison between six different implementations in terms of the number of real (complex) multipliers, real (complex) adders, twiddle factors realization, total transistor count, hardware complexity, throughput, internal wordlength, interconnect complexity and support for triple-mode, as shown in Table 15. Clearly, the proposed pipeline R4^{2}SDF-based FFT/IFFT/2D-DCT processor has the fewest complex multipliers and lowest hardware complexity, an acceptable throughput rate and moderate interconnect complexity. Although the number of the complex adders in the proposed processor is greater than the designs in [79] and [80], the total area including complex multiplier is still lower than others. The total number of transistors indicates that the proposed design achieves the smallest chip cost among architectures supporting FFT/IFFT mode.

Table 13 Hardware Cost Comparisons of the Pipelined FFT/IFFT Architecture.

Pipeline architecture	Mult. complexity	Complex Mult.	Complex adders (including constant mult.)	Complex Memory Size	Equivalent area in 256 points
R2SDF	Radix-2	$\log_2 N-2$	$2\log_2 N$	N-1	647.5
R4SDF	Radix-4	$\log_4 N-1$	$8\log_4 N$	N-1	513.5
R8SDF	Radix-8	$\log_8 N-1$	$(24+2T)\log_8 N$	N-1	617.5
$R2^2SDF$	Radix-2^2	$\log_4 N-1$	$4\log_4 N$	N-1	497.5
$R2^3SDF$	Radix-2^3	$2(\log_8 N-1)$	$6\log_8 N$	N-1	655.5
R2MDC	Radix-2	$\log_2 N-2$	$2\log_2 N$	1.5N-2	812.6
$R2^2MDC$	Radix-2^2	$\log_2 N-2$	$2\log_2 N$	1.5N-2	812.6
R4MDC	Radix-4	$3\log_4 N-3$	$4\log_2 N$	2.5N-4	1308.8
R8MDC	Radix-8	$7\log_8 N-7$	$(24+2T)\log_8 N$	4.5N-8	2673.2
Proposed $R4^2SDF$	Radix-4^2	$\log_{16} N-1$	$(8+T)\log_{16} N$	N-1	415.5

Table 14 Hardware Utilization Rate Comparisons of the Pipelined FFT/IFFT Architecture.

Pipeline architecture	Utilization rate of complex Mult.	Utilization rate of complex adders (including constant mult.)	Utilization rate of complex memory	Throughput (Output/Cycle)
R2SDF	50%	50%	100%	1
R4SDF	75%	25%	100%	1
R8SDF	87.5%	12.5%	100%	1
$R2^2SDF$	75%	50%	100%	1
$R2^3SDF$	87.5%	50%	100%	1
R2MDC	50%	50%	50%	2
$R2^2MDC$	37.5%	50%	50%	2
R4MDC	25%	25%	25%	4
R8MDC	12.5%	12.5%	12.5%	8
Proposed $R4^2SDF$	87.5%	56.25%	100%	1

Table 15 Hardware Requirement Comparison of 8×8 2D DCT Architecture.

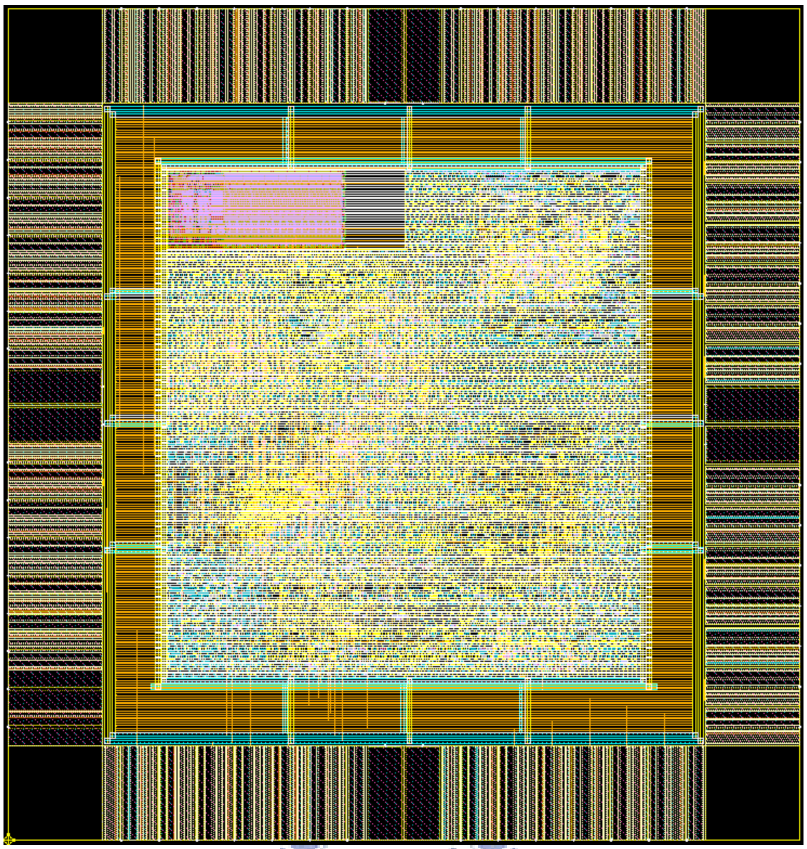
8×8 DCT	**Lee et al. [78] (parallel)**	**Chang & Wang [81] (2D systolic)**	**Hsiao and Shiue [79] (linear-array)**	**Ruetz et al. [80] (linear-array)**	**Madisetti et al. [82] (parallel MAC)**	**Proposed ($R4^2SDF$)**
Real multipliers	28	64	-	-	-	-
Real adders	134	88	-	-	-	-
Complex multipliers	-	-	3	8	14	1
Complex adders	-	-	9	18	32	26
Twiddle factors realization	Hardwired Multiplier	Hardwired Multiplier	ROM based LUT	ROM based LUT	Hardwired Multiplier	Hardwired Multiplier & ROM based LUT
Total transistor count	~ 400 K	~ 340 K	~ 105 K	N/A	~ 67 K	~60 K
Hardware complexity	O(NlogN)	$O(N^2)$	O(logN)	O(logN)	$O(\log_8 N)$	$O(\log_{16} N)$
Throughput (Output/cycle)	16	8	2	2	4	2

Internal Wordlength	18	16	16	14	22	13
Interconnect complexity	Complex	Simple	Moderate	Moderate	Simple	Moderate
FFT/IFFT/2-D DCT triple modes	No	No	No	No	No	Yes

[1] A gate count was determined and the number of transistors was determined by assuming four transistors per gate.
[2] An unknown gate count was indicated by "N/A"

6.4.3 Chip Implementation

Following the functional verification in the Matlab environment, the 256-point FFT/IFFT/2-D DCT architecture in which the internal word length of the entire design is 13-bit was synthesized by the Design Compiler with TSMC 0.13μm CMOS technology. The floorplan and post-layout were performed by Astro. The post-simulation was issued by NC-Simulator to verify the functionality after back-annotation was performed from the Start-RC extractor. The static timing check can be signed-off by PrimeTime. Finally, the power analysis and DRC were conducted using Astro Rail and Dracula, respectively. The core area of the post layout was 0.6mm^2. The reported equivalent gate count is 60086 gates, which approaches 60k gates. The gate count usage for each building block is listed in Table 16. It is obviously that 264 words shift register dominates the chip cost of 54.58%. The implementation result without the 2D DCT indicates that the total gate count decreased to 55.2k.The implementation reports in this study reveal that the routing cost penalty incurred by the additional 8×8 2D DCT mode is small. The chip operated at 100MHz, thus satisfying the high throughput requirement After the conversion, the proposed R4^{2}SDF design in 8×8 2D DCT mode could provide high frame rates of 505 kfps and 1042 kfps for frame sizes of 176×144 and 128×96 (pixel^2), respectively. Concerning the speed performance, because the pipelined multiplier operation is easy to design at a clock rate of 100 MHz or even higher, the proposed architecture can achieve a high clock rate by simple pipelining techniques for the involved arithmetic components. The chip properties shown in Fig.6.9 demonstrate that the average power dissipation of the 256-point FFT/IFFT/2-D DCT design was 22.37mW@100 MHz at 1.2V supply voltage. The layout view as shown in Fig. 39 has 64 I/O pins, of which eight pins are power supply pins. The proposed R4^{2}SDF based 256-point FFT/IFFT/2-D DCT implementation not only satisfies the system performance of DV standards in 8×8 2D DCT mode, but also achieves the satisfactory results with 40dB performance in 256-point FFT/IFFT modes. Additionally, the proposed R4^{2}SDF based 256-point FFT/IFFT/2D DCT implementation has a low power consumption (22.37 mW), and the lowest hardware requirement of all pipeline architectures. These findings indicate that the proposed design is suitable for the highly cost-efficient FFT/IFFT/2-D DCT triple-mode RSoCs IP for next-generation handheld devices.



Mode Selection	256-point FFT/IFFT and 8×8 2D-DCT
Architecture	R4^2SDF pipeline
Technology	0.13 μm CMOS
Core Size	807(μm) x 754(μm) = 0.6 mm^2
Power Consumption / Freq.	22.37 mW / 100 MHz
Accuracy / internal wordlength	40dB in DV standard / 13-bits
Input/Output/Power Pins #	29 / 27 / 8

Fig. 39: The layout view and design characteristics of proposed pipeline 256-point FFT/IFFT/8×8 2D DCT processor.

Table 16 The Gate Count Usage of Each Building Blocks.

Categories	Control	Butterfly Cores	Complex Multiplier	Constant Multipliers	Shift Registers
Area	1.3 %	21.74 %	18.9 %	3.48 %	54.58 %

6.5 Summary

This investigation develops a triple-mode reconfigurable pipeline R4^{2}SDF VLSI architecture that supports the 256-point FFT/IFFT and 8×8 2-D DCT computations. The comparison results demonstrate that the proposed R4^{2}SDF pipeline FFT/IFFT architecture has a lower hardware cost and higher utilization than R2^{2}SDF and other pipeline architectures. Following the fixed-point analysis the proposed 256-point FFT/IFFT/8×8 2-D DCT chip design is successfully implemented in 0.13μm CMOS technology with an internal wordlength of 13 bits. This design has a power consumption of 22.37 mW@100 MHz at 1.2V supply voltage. These features ensure that the proposed reconfigurable processor design is certainly amenable to the next-generation mobile communications. The upcoming fourth-generation wireless system requires the simultaneous application of many computing algorithms including MPEG-4 AVC [83] and Walsh transform [84], in the same handheld device. The reconfigurable hardware core for supporting more transforms is a significant topic for future work.

Chapter 7 Conclusion and Future Work

In this thesis, we focus on the specific ASIC design for the effective pipeline FFT/IFFT processor. Considering the hardware-orientated architecture for most efficiency, the specific FFT/IFFT processor not only minimizes the computation complexity and area cost, but also increase the hardware utilization rate with an appropriate throughput rate for different applications. For the purpose of demonstrating the effective computations in different real-time applications, four different standards have been considered, which include DTMF [12-15], MIMO-OFDM WLAN [22, 23], DVB-T [27, 28] and next generation mobile multimedia standards [5-7, 44].

For the high channel density DTMF systems, one new recursive DFT/IDFT algorithm and architecture based on a hybrid of input strength reduction scheme, the Chebyshev polynomial and register-splitting scheme is devised in this framework. The analyzed results show that the proposed VLSI algorithm leads to the fewest computation cycle and the highest throughput rate. Moreover, the proposed 212/106-point recursive DFT/IDFT chip design has been successfully implemented in 0.13 μm CMOS technology and possesses the power-efficiency consumption of 9.77 uW@20 MHz at 1.2V supply voltage for each channel. These features guarantee that the proposed high-throughput and power-efficient VLSI architecture is amenable to high channel density DTMF systems.

For the MIMO-OFDM system, we proposes a hardware-orientated approach for high efficiency to minimize the complex multiplicative complexity, area cost and achieve 100% butterfly utilization with an appropriate throughput rate. By adopting the proposed R8-FFT unit combined with the MAW method, two efficient serial blockwise type 64-point FFT/IFFT processors are constructing for the 2×2 and 4×4 MIMO-OFDM WLAN systems. For the 2×2 MIMO-OFDM system, the proposed R28MDF design has the best performance in terms of lowest complex multiplicative complexity, appropriate throughput rate of 2R, highest butterfly utilization and the fewest complex multipliers, when compared with other existing 64-point FFT/IFFT processor architectures. For the 4×4 MIMO-OFDM system, the proposed R28MDC outperforms existing FFT/IFFT pipeline processor architectures and has the lowest complex multiplicative complexity, an appropriate throughput rate of 4R, highest utilization rate (100%) of all components and the lowest hardware cost. According to the IEEE 802.11n standard [23], execution time for the 128-point and 64-point FFT/IFFT processor with 1–4 simultaneous data sequences must be calculated within 3.6 or 4.0 μs. In total, eight operational modes of the FFT/IFFT processor are required in the IEEE 802.11n standard. The effective reconfigurable FFT/IFFT processor [73] supports eight operational modes in the IEEE 802.11n standard, consumes small hardware and little power, is easily reused, and is an important topic for future work.

For the long-length based FFT computations, we develops two high effective R4^{2}SDF and R4^{3}SDF pipeline VLSI architectures that support the long-length FFT/IFFT computations. The proposed R4^{3}SDF pipeline FFT/IFFT architecture has lower multiplicative complexity and higher hardware utilization rate with smaller cost than R4^{2}SDF and other pipeline architectures. Following with fixed-point analysis in 40dB AWGN environment, the proposed R4^{2}SDF and R4^{3}SDF based 4096-point FFT/IFFT designs are successfully implemented in 0.13 μm CMOS technology with an internal word-length of 14 and 13-bits, respectively. The proposed R4^{2}SDF and R4^{3}SDF based design have a low power consumption of 6.3725 and 5.985 mW @20 MHz at 1.2V supply voltage. Thus, these features ensure that the proposed R4^{3}SDF pipeline processor design certainly meets the high effective VLSI architecture.

For the next-generation mobile applications, we develops a triple-mode reconfigurable pipeline R4^{2}SDF VLSI architecture that supports the 256-point FFT/IFFT and 8×8 2-D DCT computations. The comparison results demonstrate that the proposed R4^{2}SDF pipeline FFT/IFFT architecture has a lower hardware cost and higher utilization than R2^{2}SDF and other pipeline architectures. Following the fixed-point analysis the proposed 256-point FFT/IFFT/8×8 2-D DCT chip design is successfully implemented in 0.13μm CMOS technology with an internal wordlength of 13 bits. This design has a power consumption of 22.37 mW@100 MHz at 1.2V supply voltage. These features ensure that the proposed reconfigurable processor design is certainly amenable to the next-generation mobile communications. The upcoming fourth-generation wireless system requires the simultaneous application of many computing algorithms including MPEG-4 AVC [83] and Walsh transform [84], in the same handheld device. Then, the reconfigurable hardware core for supporting more transforms is a significant topic for future work. Furthermore, the fixed word-length analysis for each building block to reduce more hardware cost is also important future investigations. According to the comprehensive comparisons and implementation results, we could provide that the proposed RDFT, R28MDF/R28MDC, R4^{2}SDF/ R4^{3}SDF and Triple-Mode designs achieve the high effective advantages for DTMF, MIMO-OFDM WLAN, DVB-T and next-generation applications.

Bibliography

[1] W. W. Smith, J. M. Smith, *Handbook of Real-Time Fast Fourier Transforms*. Piscataway, NJ: IEEE Press, 1995.

[2] E. Oran Brigham, *The Fast Fourier Transform*, Prentice-Hall, Englewood Cliffs, NJ, 1974.

[3] J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series, "*Mathematics of Computation*, Vol. 19, p. 297, 1965.

[4] G. Goertzel, "An algorithm for the evaluation of finite trigonometric series," *American Math. Monthly*, vol. 65, pp. 34-35, Jan. 1958.

[5] S. M. Chai, S. Chiricescu, R. Essick, B. Lucas, P. May, K. Moat, J. M. Norris and M. Schuette," Streaming Processors for Next-Generation Mobile Imaging Application," *IEEE Comm. Mag.*, vol. 43, issue 12, pp. 81-89, Dec. 2005.

[6] R. K. Kolagotla, J. Fridman, M. M. Hoffiman, W. C. Anderson, B. C. Aldrich, D. B. Witt, M. S. Allen, R. R. Dunton and L. A. Booth, " A 333-MHz dual-MAC DSP architecture for next-generation wireless application," *IEEE Inter. Conf. on Acou., Speech, and Signal Proc.*, vol. 2, pp. 1013-1016, May 2001.

[7] M. Vorbach and J. Becker, "Reconfigurable processor architectures for mobile phones," *IEEE Inter. Symp. Parallel and Distributed Proc.*, 22-26, Apr. 2003.

[8] E. Tell, O. Seger and D. Liu, "A converged hardware solution for FFT, DCT and Walsh transform," *IEEE Inter. Symp. Signal Proc. and its Applications*, vol. 1, pp. 609-612, July 2003.

[9] R. Storn, "Efficient input reordering for the DCT based on a real-valued decimation-in-time FFT," *IEEE Signal Proc. Letters*, vol. 3, no. 8, pp. 242-244, Aug. 1996.

[10] C. Diab, M. Oueidat and R. Prost, "A New IDCT-DFT Relationship Reducing the IDCT Computational Cost," *IEEE Trans. On Signal Proc.*, vol. 50, no. 7, pp. 1681-1684, July 2002.

[11] ITU Blue Book, Recommendation Q. 24: Multi-Frequency Push-Bottom Signal Reception, Geneva, Switzerland, 1989.

[12] S. L. Gay, J. Hartung, and G. L. Smith, "Algorithms for muti-channel DTMF detection for the WE DSP32 family," *in Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 1134-1137, Apr. 1989.

[13] M. D. Felder, J. C. Mason, and B. L. Evans, "Efficient dual-tone multifrequency detection using the nonuniform discrete Fourier transform," *IEEE Signal Processing Lett.*, vol. 5, pp. 160-163, Jul. 1998.

[14] J. P. Min, J. L. Sang and H. Y. Dal, "Signal detection and analysis of DTMF detector with quick Fourier transform," *The 30th Annual Conf. of the IEEE Industrial Electronics Society*, pp. 2058-2064, Nov. 2004.

[15] D. Vanzquez, M. J. Avedillo, G. Huertas, J. M. Quintana, M. Pauritsh, A. Rueda and J. L. Huertas, "A low-voltage low-power high performance fully integrated DTMF detector," *IEEE International Solid-State Circuit Conf.* , pp. 353-356 Sep. 2001.

[16] Conferencing chip specification, High-density conference meeting for the telephone systems, ADT Inc. Available: http: //www.adaptivedigital.com/pdf/adt_conf_c64x_chip.pdf

[17] Texas Instruments technical white paper, Carrier Class, High Density VoP white Paper, Jan, 2001, Available: http://focus.ti.com/lit/ml/spey003/spey003.pdf.

[18] Voice over Packet Processor Product Specification, AC491xxx High Density Voice over Packet Processor Family, AudioCodes Inc. Available: http://www.audiocodes.com/Objects/LTRT-00270_DS_AC 491.pdf.

[19] Voice Gateway Product Specification, Single and High-Density Voice over IP Support for the Ciso AS5300/Voice Gateway, Cisco Inc. Available: http://www.cisco.com/warp/public/cc/pd/as/as5300/prodlit/vffc_ds.pdf.

[20] M. Ding, Z. Shen, and B. L. Evans, "An achievable performance upper bound for discrete multitone equalization," *IEEE Global Telecommunications Conf.*, vol. 4, pp. 2297-2301, Dec. 2004.

[21] R. K. Martin, K. Vanbleu, M. Ding, G. Yebaert, M. Milosevic, B. L. Evans, M. Moonen and C. R. Johson, Jr., " Unification and evaluation of equalization structures and design algorithms for discrete multitone modulation systems," *IEEE Trans. Signal Processing*, vol. 53, no. 10, pp. 3880-3894, Oct. 2005.

[22] R. V. Nee and R. Prasad, "OFDM for wireless multimedia communications, " Norwood, MA: Artch House, 2000.

[23] Mujtaba et al.: 'TGn Sync Proposal Tech. Specification for IEEE 802.11 Task Group 2005', *IEEE 802.11-04/0889r3*, 2005

[24] D. Borkowski, and L. Bruhl,: "Optimized hardware architecture for real-time equalization in single- and multi-carrier MIMO systems, " *Proc. 3rd Workshop on*

Software Radio, Karlsruhe, Germany, 2004.

[25] S. Ludwig and Z. Ernst: “Optimized FFT architecture for MIMO application,” *Proc. 13th European Signal Processing Conference*, Antalya, Sep. 2005.

[26] T. Sansaloni, A. Perez-Pascual, V. Torres and J. Valls, “Efficient pipeline FFT processors for WLAN MIMIO-OFDM systems,” *IEE Electronics Letters*, vol. 41, issues 19, pp.1043-1044, Sep. 2005.

[27] ETSI, “Digital Video Broadcasting (DVB): Transmission System for Handheld Terminals (DVB-H),” *ETSI EN302304*.

[28] C. T. Lin and Y. C. Yu, “Cost-Effective Pipeline FFT/IFFT VLSI Architecture for DVB-H System,” *National Symp. on Telecommunication*, pp. 295-299, Nov. 2007.

[29] C. L. Wang and C. H. Chang, “A new memory-based FFT processor for VDSL transceivers, “ *IEEE Inter. Symp. on Circuits and System*, vol. 4, pp. 670-673, May 2001.

[30] M. Jun, Y. Yahat and K. Yamaguchi, “A study on annoyance of musical signal using LAeq measurement and digital signal processing, “ *IEEE Inter. Conf. on Acoustics, Speech and Signal Proc.*, vol. 11, pp. 1281-1284, Apr. 1986.

[31] S. He and M. Torkelson, “Designing pipeline FFT processor for OFDM (de)modulation, “ *in Proc. URSI Int. Symp. Signals, Syst., Electron.*, pp. 257-262, 1998.

[32] W. C. Yeh and C. W. Jen, “High-speed and low-power split-radix FFT,” *IEEEE Trans. on Signal Processing*, vol. 51, no. 3, pp. 864-874, Mar. 2003.

[33] K. Maharatna, E. Grass and U. Jaghold, “A 64-point Fourier transform chip for high-speed wireless LAN application using OFDM,” *IEEE J. Solid-State Circuits*, vol. 39, issue 3, pp. 484-493, Mar. 2004.

[34] W. H. Chang and T. Nguyen, “An OFDM-specified lossless FFT architecture, “ *IEEE Trans. on Circuits and Systems I*, vol. 53, issue 6, pp. 1235-1243, June 2006.

[35] L. Jia, Y. Gao and H. Tenhunen, “ Efficient VLSI implementation of radix-8 FFT algorithm,” *IEEE Pacific Rim Conf. on Comm., Computers and Signal Proc.*, pp. 468-471, Aug. 1999.

[36] Y. Jung, Y. Tak, J. K. J. Park, D. Kim and H. Park, “Efficient FFT Algorithm for OFDM Modulation”, *IEEE Int. Conf. on Electrical and Electronic Tech.* , vol. 2, pp.

676-678, Aug. 2001.

[37] W. Li and L. Wanhammar, “A pipeline FFT processor,” *in Proc. IEEE Workshop on Signal Processing System*s, pp. 654-662, 1999.

[38] A. P. Chandrakasan and R. W. Brodersen, “Low Power Digital CMOS Design”, Kluwer Academic Publishers, 1995.

[39] L. R. Rabiner and B. Gold, “Theory and Application of Digital Signal Processing”, Prentice-Hall, Inc., NJ, 1975.

[40] E. E. Swatzlander, W. K. W. Young and S. J. Joseph, “A radix 4 delay commutator for fast Fourier transform processor implementation”, *IEEE J. Solod-State Circuits*, SC-19(5), pp. 702-709, Oct. 1984.

[41] K. Maharatna, E. Grass and U. Jaghold, “A 64-point Fourier transform chip for high-speed wireless LAN application using OFDM,” *IEEE J. Solid-State Circuits*, vol. 39, issue 3, pp. 484-493, Mar. 2004.

[42] E. H. Wold and A. M. Despain, “Pipeline and parallel pipeline FFT processors for VLSI implementation, “ *IEEE Trans. Comput.*, C-33, pp. 414-426, May 1984.

[43] A. M. Despain, “Fourier transform computer using CORDIC iterations,” *IEEE Trans. Comput.*, , C-23, pp. 993-1001, Oct. 1974.

[44] IEEE Standard 802.16-2004,” IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Fixed Broadband Wireless Access Systems,” New York: IEEE, 2004.

[45] Z. Liu, Y. Song, T. Ikenaga and S. Goto, “A VLSI array processing oriented fast Fourier transform algorithm and hardware implementation,” *IEICE Trans. Fundamentals*, vol.E88-A, no. 12, pp. 3523-3530, Dec. 2005.

[46] Z. Wang, G. A. Jullien and W. C. Miller, “Recursive algorithms for the forward and inverse discrete cosine transform with arbitrary length,” *IEEE Signal Processing Lett.*, vol. 1, no. 7, pp. 101-102, Jul. 1994.

[47] C. H. Chen, B. D. Liu, J. F. Yang, and J. L. Wang, “Efficient recursive structures for forward and inverse discrete cosine transform,” *IEEE Trans. Signal Processing*, vol.

52, pp. 2665-2669, Sep. 2004.

[48] M. F. Aburdene, J. Zheng and R. J. Kozick, “Computation of discrete cosine transform using Clenshaw’s recurrence formula,” *IEEE Signal Processing Lett.*, vol. 2, no. 8, pp. 155-156, Aug. 1995.

[49] V. V. Cizek, “Recursive calculation of Fourier transform of discrete signal,” I*EEE Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 28-31, May 1982.

[50] T. E. Curtis and M. J. Curtis, “Recursive implementation of prime radix and composite radix Fourier transforms,” *IEE Colloquium on Signal Processing Applications of Finite Field Mathematics*, pp. 2/1-2/9, Jun. 1989.

[51] L. D. Van and C. C. Yang, "High-speed area-efficient recursive DFT/IDFT architectures," *in Proc. IEEE Int. Symp. Circuits Syst.*, vol. 3, pp. 357-360, May 2004.

[52] J. F. Yang and F. K. Chen, “Recursive discrete Fourier transform with unified IIR filter structures,” *Elsevier Science B.V., Signal Processing*, vol. 82, pp. 31-41, Jan. 2002.

[53] C. P. Fan and G. A. Su, “Novel recursive discrete Fourier transform with compact architecture,” *IEEE Asia-Pacific Conf. Circuits Syst.*, pp. 1081-1084, Dec. 2004.

[54] L. D. Van, Y. C. Yu, C. M. Huang, C. T. Lin, "Low computation cycle and high speed recursive DFT/IDFT: VLSI algorithm and architecture," *in Proc. IEEE Workshop on Signal Processing Systems (SiPS)*, pp. 579-584, Nov. 2005.

[55] L. D. Van, C. T. Lin and Y. C. Yu, “VLSI architecture for the low-computation cycle and power-efficient recursive DFT/IDFT Design,” *IEICE Trans. Fundamentals*, vol.E90

[56] C.C.W. Hui, T.J. Ding, and J. V. McCanny, “A 64-point Fourier transform chip for video motion compensation using phase correlation,” *IEEE J. Solid-State Circuits*, , vol. 31, issues 11, pp. 1751-1761, Nov. 1996.

[57] C. T. Lin, Y. C. Yu and L. D. Van, “A Low Power 64-Point FFT/IFFT Design for IEEE 802.11a wireless LAN Application,” *Proc. IEEE Int. Symp. On Circuits and System*, pp. 4523-4526, May. 2006.

[58] E. Cornu, N. Destrez, A. Dufaux, H. Sheikhzqadeh and R. Brennan, “An ultra low power, ultra miniature voice command system based on hidden markov models,” *IEEE Inter. Conf. on Acoustics, Speech, and Signal Proc.*, vol. 4, pp. 3800-3803, May 2002.

[59] S. Bouguezel, M. O. Ahmad, and M. N. S. Swamy, “A New Radix-2/8 FFT Algorithm for Length-q x 2^m DFTs,” *IEEE Trans. On Circuits and Systems I*, vol. 51,

pp.1723-1732, Sep. 2004.

[60] L. Jia., Y. Gao, J. Isoaho, and H. Tenhunen," A new VLSI-oriented FFT algorithm and implementation," *Proc. Eleventh Annu. IEEE Int. ASIC Conf.*, pp. 33-341, 1998.

[61] Y. Jung, H. Yoon and K. Jaeseok, "New efficient FFT algorithm and pipeline implementation results for OFDM/DMT applications," *IEEE Trans. on Consumer Electronics*, vol. 49, issues 1, pp. 14-20, Feb. 2003.

[62] A. V. Opppenheim and R. W. Schafer, *Discrete-Time Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1989.

[63] P. Duhamel and H. Hollmann, "Split-radix FFT algorithm," *Electronic Letters,* vol. 20, No. 1, pp. 14-16, Jan., 1984.

[64] C. S. Burrus, "Index mapping for multidimensional formulation of the DFT and convolution, " *IEEE Trans. Acoust., Speech, Signal Processing,* ASSP-25(3): 239-242, June 1977.

[65] K. K. Parhi, "VLSI Digital Signal Processing Systems: Design and Implementation," *NY: Wiley*, 1999.

[66] L. D. Van, Y. C. Yu, C. M. Huang, C. T. Lin, "Low computation cycle and high speed recursive DFT/IDFT: VLSI algorithm and architecture," in *Proc. IEEE Workshop on Signal Processing Systems (SiPS)*, pp. 579-584, Nov. 2005.

[67] L. R. Rabiner and B. Gold, "Theory and Application of Digital Signal Processing," *NJ Prentice-Hall Inc.*, 1975.

[68] S. C. Chen, C. T. Yu, C. L. Tsai, and J. J. Tang, "A new IFFT/FFT hardware implementation structure for OFDM applications," I*EEE Asia-Pacific Conf. on Circuits and Systems*, vol. 2, pp.1093-1096, Dec. 2004.

[69] A. M. Despain, "Fourier transform computer using CORDIC iterations," *IEEE Trans. Comput.*, C-23, pp. 993-1001, Oct. 1974.

[70] G. Bi and E. V. Jones, "A pipelined FFT processor for word-sequential data," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 1982-1985, Dec. 1989.

[71] J. Garcia, J. A. Michell and A. M. Buron, "VLSI configurable delay commutator for a pipeline split radix FFT architecture," *IEEE Trans. Signal Processing*, vol. 47, pp. 3098-3107, Nov. 1999.

[72] H. Jiang, H. Luo, J. Tian and W. Song, "Design of an efficient FFT processor for OFDM systems," *IEEE Trans. On Consumer Electronics*, vol. 51, pp. 1099-1103, Nov. 2005.

[73] Y. W. Lin and C. Y. Lee, "Design of an FFT/IFFT Processor for MIMO OFDM

Systems," *IEEE Trans. On Circuits and Systems I*, vol. 54, issues 4, pp. 807-815, Apr. 2007.

[74] S. F. Hsiao, Y. H. Hu, T. B. Juang and C. H. Lee, "Efficient VLSI Implementations of Fast Multiplierless Approximated DCT Using Parameterized Hardware Modules for Silicion Intellectual Property Design," *IEEE Trans. on Circuits and Systems I,* vol. 52, no. 8, pp. 1568-1579, Aug. 2005.

[75] "DVCAM format overview," Sony, http://www.sony.ca/dvcm/brochures.htm.

[76] A. Silva, P. Gouveia, and A. Navarro, "Fast multiplication-free QWDCT for DV coding standard, " *IEEE Trans. on Consumer Elec.*, vol. 50, no. 1, Feb. 2004.

[77] A. Ichigaya, M. Kurozumi, N. Hara, Y. Nishida and E. Nakasu, "A method of estimating coding PSNR using quantized DCT coefficients", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 2, Feb. 2006.

[78] Y. P. Lee, T. H. Chen, L. G. Chen, M. J. Chen, and C. W. Ku, "A cost effective architecture for 8×8 two-dimensional DCT/IDCT using direct method," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 459-467, June 1997.

[79] S. F. Hsiao and W. R. Shiue, " A new hardware-efficient algorithm and architecture for computation of 2-D DCTs on a linear array", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 11, pp. 1149-1159, Nov. 2001.

[80] P. A Ruetz, P. Tong, D. Bailey, D. A. Luthi, and P. H. Ang, "A high performance full-motion video compression chip set, " *IEEE Trans. Circuits Syste. Video Technol.*, vol. 2, no. 2, pp. 111-121, June 1992.

[81] Y.-T. Chang and C.-L. Wang, "New systolic array implementation of the 2-D discrete cosine transform and its inverse," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, pp. 150-157, Apr. 1995.

[82] A. Madisetii and A. N. Willson, "A 100 MHz 2-D 8×8 DCT/IDCT processor for HDTV application, " *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, no. 2, pp. 158-164, Apr. 1995.

[83] G. A. Jian, C. D. Chien and J. I. Guo, "A memory-based hardware accelerator for real-time MPEG-4 Audio Coding and Reverberation," *IEEE Inter. Symp. on Circuit and Syst.*, pp. 1569-1572, May 2007.

[84] R. Pandey and M. L. Bushnell, "Architecture for variable-length combined FFT, DCT and MWT transform hardware for multi-modeWireless system," *IEEE Inter. Conf. on Embedded Syst.*, pp. 121-126, Jan. 2007.

Appendix A

The Transfer Function between 8×8 2D SFFT and 8×8 2D DCT

Equation (90) reveals that the 2D DCT result of $X[k1,k2]$ can be derived from the 8×8 2D SFFT with a time-domain shift of 1/4 samples.

$$
\begin{aligned}
Y_s[k_1,k_2] &= \sum_{n_1=0}^{7}\sum_{n_2=0}^{7} y(n_1,n_2)\cdot W_8^{(n_1+\frac{1}{4})k_1}\cdot W_8^{(n_2+\frac{1}{4})k_2} \\
&= \sum_{n_1=0}^{7}\sum_{n_2=0}^{7} y(n_1,n_2)\cdot[\cos(\frac{\pi k_1(1+4n_1)}{16})+j\sin(\frac{\pi k_1(1+4n_1)}{16})] \\
&\qquad \cdot[\cos(\frac{\pi k_2(1+4n_2)}{16})+j\sin(\frac{\pi k_2(1+4n_2)}{16})]
\end{aligned}
$$

where $0 \le k_1, k_2, n_1, n_2 \le 7$. Because the input data $y(n1,n2)$ is a real-valued sequence, the second half output of the 8×8 2D-shifted SFFT can be calculated as

$$
\begin{aligned}
Y_s[8-k_1,k_2] &= \sum_{n_1=0}^{7}\sum_{n_2=0}^{7} y(n_1,n_2)\cdot W_8^{(n_1+\frac{1}{4})(8-k_1)}\cdot W_8^{(n_2+\frac{1}{4})k_2} \\
&= \sum_{n_1=0}^{7}\sum_{n_2=0}^{7} y(n_1,n_2)\cdot W_8^{8(n_1+\frac{1}{4})}\cdot W_8^{-(n_1+\frac{1}{4})k_1}\cdot W_8^{(n_2+\frac{1}{4})k_2} \\
&= (-j)\cdot \sum_{n_1=0}^{7}\sum_{n_2=0}^{7} y(n_1,n_2)\cdot W_8^{-(n_1+\frac{1}{4})k_1}\cdot W_8^{(n_2+\frac{1}{4})k_2} \\
&= (-j)\cdot \sum_{n_1=0}^{7}\sum_{n_2=0}^{7} y(n_1,n_2)\cdot[\cos(\frac{\pi k_1(1+4n_1)}{16})-j\sin(\frac{\pi k_1(1+4n_1)}{16})] \\
&\qquad \cdot[\cos(\frac{\pi k_2(1+4n_2)}{16})+j\sin(\frac{\pi k_2(1+4n_2)}{16})]
\end{aligned}
$$

According to Eq. (6.2) in the original manuscript, $X[k1,k2]$ can only result from the product of two real parts of the twiddle factors $W_8^{(n_1+\frac{1}{4})k_1}$ and $W_8^{(n_2+\frac{1}{4})k_2}$

$$
X[k_1,k_2] = \sum_{n_1=0}^{7}\sum_{n_2=0}^{7} y(n_1,n_2)\cdot\cos(\frac{\pi k_1(1+4n_1)}{16})\cdot\cos(\frac{\pi k_2(1+4n_2)}{16}).
$$

(8 9)

Furthermore, the real value of $Y_S(k_1,k_2)$ and the imaginary value of $Y_S(8-k_1,k_2)$ can be written as

$$\mathrm{Re}\{Y_s[k_1,k_2]\}=\sum_{n_1=0}^{7}\sum_{n_2=0}^{7}y(n_1,n_2)\cdot\{\cos(\frac{\pi k_1(1+4n_1)}{16})\cdot\cos(\frac{\pi k_2(1+4n_2)}{16})-\sin(\frac{\pi k_1(1+4n_1)}{16})\cdot\sin(\frac{\pi k_2(1+4n_2)}{16})\}$$

$$\mathrm{Im}\{Y_s[8-k_1,k_2]\}=\sum_{n_1=0}^{7}\sum_{n_2=0}^{7}y(n_1,n_2)\cdot\{\sin(\frac{\pi k_1(1+4n_1)}{16})\cdot\sin(\frac{\pi k_2(1+4n_2)}{16})-\cos(\frac{\pi k_1(1+4n_1)}{16})\cdot\cos(\frac{\pi k_2(1+4n_2)}{16})\}$$

The 8×8 2D DCT result can thus be expressed as a subtraction of the imaginary value of $Y_S(8-k_1,k_2)$ from the real value of $Y_S(k_1,k_2)$ and.

$$X[k_1,k_2]=\frac{1}{2}\{\mathrm{Re}[Y_S(k_1,k_2)]-\mathrm{Im}[Y_S(8-k_1,k_2)]\}$$

Only the real value of $Y_S(k_1,k_2)$, the imaginary value of $Y_S(8-k_1,k_2)$ and the real value of the input port $y(n1,n2)$ are adopted to obtain the single $X[k_1, k_2]$ in the 2D 8×8 SFFT based design. However, the proposed R4^{2}SDF design is a complex system. Two reordered input sequences $\{y_1(n_1,n_2)\},\{y_2(n_1,n_2)\}$ for two independent real input sequences $\{x_1(n_1,n_2)\},\{x_2(n_1, n_2)\}$ can be combined to form a complex input sequence $\{ y(n_1,n_2) = y_1(n_1,n_2) + jy_2(n_1,n_2)\}$, and the double throughput of 2D 8×8 DCT of $\{x_1(n_1,n_2)\},\{x_2(n_1,n_2)\}$ can be derived by the single 2D 8×8 SFFT computation. Consequently, two independent 8×8 2D DCTs $X_1[k_1,k_2]$, $X_2[k_1,k_2]$ of $x_1(n_1,n_2)$, $x_2(n_1,n_2)$, respectively, can then be obtained as below:

$$X_1[k_1,k_2]=\frac{1}{4}\{\mathrm{Re}[Y_s(k_1,k_2)]-\mathrm{Re}[Y_s(8-k_1,8-k_2)]\}-\frac{1}{4}\{\mathrm{Im}[Y_s(8-k_1,k_2)]+\mathrm{Im}[Y_s(k_1,8-k_2)]\},$$

$$X_2[k_1,k_2]=\frac{1}{4}\{\mathrm{Im}[Y_s(k_1,k_2)]-\mathrm{Im}[Y_s(8-k_1,8-k_2)]\}+\frac{1}{4}\{\mathrm{Re}[Y_s(8-k_1,k_2)]+\mathrm{Re}[Y_s(k_1,8-k_2)]\}.$$

Two different 8×8 2D DCT results are obtained from one single 8×8 2D SFFT computation as above.

VITA

姓名： 余遠渠

性別： 男

生日： 民國 61 年 10 月 7 日

籍貫： 台灣省桃園縣

學歷：

1. 民國 78 年 6 月苗栗市建台高級中學畢業
2. 民國 82 年 6 月私立逢甲大學自動控制工程系學士畢業
3. 民國 84 年 6 月私立逢甲大學自動控制工程系碩士畢業
4. 民國 92 年 9 月國立交通大學電機與控制工程學系博士班

經歷：

1. 民國 86 年 6 月~民國 86 年 10 月(4 個月): 楊宇科技—專案經理
2. 民國 88 年 9 月~民國 90 年 2 月(1 年 5 個月):
   宏碁電腦—數位電路設計, 資深工程師
3. 民國 90 年 2 月~民國 97 年 1 月(6 年 11 個月):
   義隆電子—數位 IC 設計,副理
4. 民國 97 年 1 月迄今(3 個月): 智微科技—數位 IC 設計,經理

PUBLICATION LISTS

期刊部分：

[1] Lan-Da Van, Chin-Teng Lin and Yuan-Chu Yu, "VLSI Architecture for the Low-Computation Cycle and Power-Efficient Recursive DFT/IDFT Design," *IEICE Trans. on Electronics, Information and Communication Engineers,* Vol. E90-A, No. 8, Aug. 2007.

[2] Chin-Teng Lin, Yuan-Chu Yu and Lan-Da Van, "Cost-Effective Triple-Mode Reconfigurable Pipeline FFT/IFFT/2-D DCT Processor, " *IEEE Trans. on VLSI,* Accepted, 2007.

[3] Chin-Teng Lin and Yuan-Chu Yu, "Design of an Effective Pipeline FFT/IFFT Processor, " *International Journal of Electrical Engineering (IJEE),* Accepted, 2008.

會議論文部分：

[1] Lan-Da Van, Yuan-Chu Yu, Chun-Ming Huang, Chin-Teng Lin, "Low computation cycle and high speed recursive DFT/IDFT: VLSI algorithm and architecture," in *Proc. IEEE Workshop on Signal Processing Systems (SiPS 2005),* Nov. 2005, pp.

579-584, Athens, Greece.

[2] Chin-Teng Lin, Yuan-Chu Yu, and Lan-Da Van, “A Low Power 64-Point FFT/IFFT Design for IEEE 802.11a WLAN Application”, *IEEE Int. Symp. on Circuits and Syst. 2006 (ISCAS 2006)*, May 21-24, 2006.

[3] Chin-Teng Lin, and Yuan-Chu Yu “Cost-Effective Pipeline FFT/IFFT VLSI Architecture for DVB-T System”, *National Symp. on Telecommunications 2007 (NST 2007)*, Oct. 28, Taipie, Taiwan.

申請之專利部分：

[1] Chin-Teng Lin, Lan-Da Van and Yuan-Chu Yu, “可正反轉共用之遞迴式離散傅力葉處理器單核心裝置及其應用, “*0007-158,* 公告中, 2007.

[2] Chin-Teng Lin, Yuan-Chu Yu and Lan-Da Van, “快速傅立葉轉換及其反轉換裝置與方法, “*096135456,* 公告中, 2007.