A Hardware-efficient Variable-length FFT Processor for Low-power Applications

Yifan Bo*, Renfeng Dou*, Jun Han* and Xiaoyang Zeng*
*State-Key Lab of ASIC and System, Fudan University, Shanghai, 200433, China.
E-mail: {11212020001, junhan}@fudan.edu.cn

Abstract—**The fast Fourier transformation (FFT) is a key operation in digital signal processing (DSP) systems and has been studied intensively to improve the performance. Nowadays, embedded DSP systems require low energy consumption to prolong the life cycle, which raises stringent power limitation for FFT processing. Meanwhile, sufficient signal-to-quantization-noise ratio (SQNR) is a basic requirement in these systems. In this paper, a modified data scaling scheme as well as trounding method is employed to improve the SQNR performance. Therefore word-length can be reduced and energy is saved accordingly. Memory-based architecture is chosen to support variable-length FFT processing. Also, constant multiplier array is introduced in the datapath to reduce the power dissipation with a slight increase of area. The proposed processor can perform 64 - 8192-point FFT processing. The core area is 2.29 mm^2 and the power consumption is 67.9 mW at 100MHz. Besides, the SQNR of 55.4 dB and 33.3 dB are achieved for 64-point and 8192-point FFT respectively.**

I. Introduction

In recent years, researches on wearable and implanted medical devices have attracted more and more attention. These devices share a common feature of low energy consumption in order to prolong their life cycle. As a result, low power or even ultra-low power design techniques become more and more crucial, especially in biomedical systems and wireless micro-sensor networks [1].

Many biomedical signal processing systems and methods have been presented to deal with various signals for different applications. Karlen classifies sleep/wake states based on cardiorespiratory signals [2]. Tseng analyses heart rate variability (HRV) [3], which is an indicator of cardiovascular health. A system for biomedical spectrum analysis is proposed by Nie [4]. Among those systems and methods, the fast Fourier transform (FFT), converting signal from the time domain to the frequency domain, is used for biosignal analysis and can be one of the most energy consuming tasks. Reducing the data word-length for FFT decreases the energy consumption but leads to a loss of signal-to-quantization-noise ratio (SQNR). Therefore, one challenge for biomedical signal systems design is to lower the energy consumption of FFT while preserving reasonable SQNR. Of course, low energy consumption and high SQNR are also required by many other applications of FFT, such as orthogonal frequency division multiplexing (OFDM) systems for digital communication [5] [6]. In particular, OFDM systems need the FFT implementation that uses a shorter word-length as well as meets both SQNR and throughput requirements [7].

In this paper, we propose a low-power FFT design considering both algorithm and architecture. Compared to the traditional scaling method [8], the proposed modified method uses four scaling factors after the first stage of FFT computation based on radix-4 algorithm. Therefore this method has a distinct advantage for signals with a large crest factor, i.e., the ratio between peak and mean value of the input, like ECG signals. Also, trounding strategy [9], a compromise between truncation and rounding, is employed to improve the SQNR performance. Besides, we implement a tailored constant multiplier array within the butterfly unit. An index determination mechanism is introduced to decide whether the butterfly unit uses the constant multiplier array or complex multipliers during the computation. Thus, the power consumption of datapath is reduced with a slight increase in total number of gates.

The rest of this paper is organized as follows. Section II describes the methods aiming at improving SQNR performance, including modified data scaling mechanism and the trounding strategy. The overall architecture of FFT processor and the design of tailored constant multiplier array are presented in Section III. Section IV summarizes and compares the performance with other works. Finally, the conclusion is given in Section V.

II. FFT Algorithm

For a given sequence $x(n)$, the N-point discrete Fourier transform (DFT) is defined as

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

W_N^{nk} is known as the twiddle factor, that is

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

The direct implementation of (1) has a computational complexity of $O(N^2)$. By using the symmetry and periodicity properties of the twiddle factors [10], the FFT algorithm can reduce the complexity to $O(N\log_r N)$, where r is the radix of FFT processing. Radix-4 implementation is widely used because it can save about 25% of total complex multiplications compared to radix-2 computation and the hardware cost is much less than higher radix implementation. Meanwhile, according to Chang's work [11], the decimation in frequency (DIF) algorithm has better SQNR performance than that of decimation in time (DIT) because most of nontrival twiddle

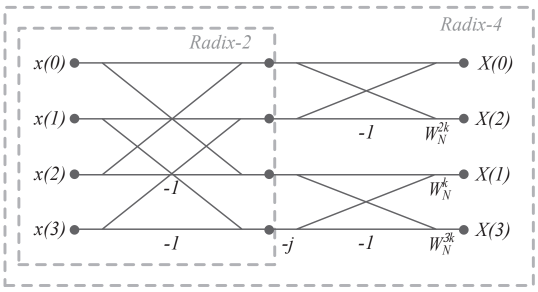

Fig. 1. Radix-4/2 mixed-radix butterfly.

factors are concentrated in the later stages of FFT processing. Thus, this paper employs a radix-4/2 DIF butterfly unit. Mixed-radix is used to cover all 2^n-point FFTs. As shown in Fig. 1, the butterfly unit can calculate either one radix-4 computation or two radix-2 operations.

A. *Modified Data Scaling Method*

Overflows and round-off errors in the arithmetic units lead to the degradation of accuracy. Obviously, overflows produce more severe impairment since the number loses its most significant bits. In contrast, round-off errors cause lost in the least significant bits. Thus, data scaling methods are widely used in prior works to prevent overflows so as to improve the SQNR performance [1] [12] [13]. Generally, the scaling methods employ one or a few scaling factors for a whole stage of FFT processing. Ickes's work [1] uses just one scaling factor in a FFT stage. It is determined by the maximum value of the intermediate results in the stage. The accuracy of this method will be strongly affected when dealing with signals having a large crest factor. It is because the maximum value after computation can be several times larger than the rest values. Therefore, in order to scale the maximum value, grave round-off errors will occur when scaling the rest values. The dynamic scaling approach in [12] is based on a prefetch buffer that divides a whole stage's complex numbers into several blocks. Each block has a scaling factor so as to reduce the round-off errors. However, this method requires many internal storage elements as well as extra reshuffle operations between stages.

The proposed modified data scaling method employs four scaling factors after the first stage of FFT computation. Compared to the traditional scaling mechanism [1], only three additional comparators and a few registers are needed to support our method. No extra storage element or reshuffle operation is required. As shown in Fig. 2, take 64-point FFT processing as an example. The butterfly computation in the first stage has four outputs, each of which is located in different blocks. Therefore, after the first stage, the intermediate data can be divided into four blocks, i.e., $[0, \frac{N}{4}-1]$, $[\frac{N}{4}, \frac{N}{2}-1]$, $[\frac{N}{2}, \frac{3N}{4}-1]$, and $[\frac{3N}{4}, N-1]$, where N means the transform size. Since these four blocks are independent of each other during the sequential computation, it is reasonable to use one scaling factor within each block. Data are scaled according to corresponding scaling factor when operated during the later stages of FFT processing.

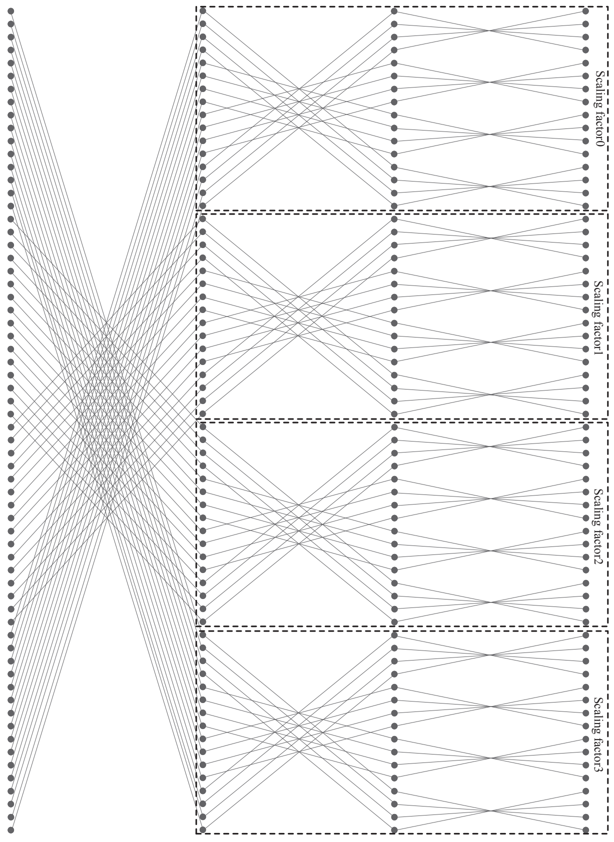

Fig. 2. Modified data scaling method for 64-point FFT processing.

B. *Trounding Strategy*

The quantization loss is unavoidable in DSP systems according to the effect of finite word-length. During FFT processing, the product of multiplier usually requires twice as many bits as the input number. Quantization loss occurs in the multiplication operations in the butterfly unit if the product is represented with the original word-length of input. Generally, truncation and rounding are the two ways to shorten the word-length. Truncation means simply removing several least significant bits (LSBs). In contrast, rounding is a more accurate method since it truncates the number after an addition operation. However, the additional adder will increase the critical path delay and energy consumption. Trounding strategy [9] turns out to be a compromise between truncation and rounding. As shown in Fig. 3, an OR gate is employed so that trounding

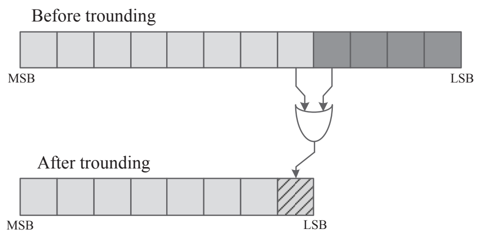

Fig. 3. Trounding strategy.

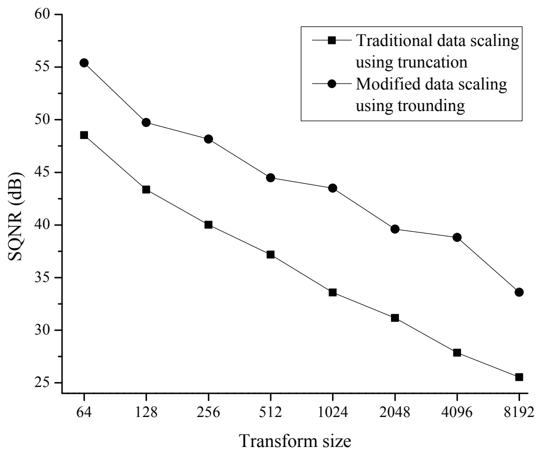

Fig. 4. SQNR comparison between traditional data scaling using truncation and modified data scaling using trounding.

preserves information from the truncated part though it only affects the LSB. Its accuracy can be approximately equal to the rounding method with little hardware cost.

Fig. 4 shows the SQNR comparison between traditional data scaling using truncation and modified data scaling using trounding. Up to 9 dB SQNR improvement is achieved by employing trounding and the proposed modified data scaling method using ECG signal inputs. Thus, less word-length is needed since our method can provide higher SQNR, which helps lower the overall energy consumption.

III. Architecture of FFT Processor

A. Architectural Overview

The overall architecture of proposed FFT processor is shown in Fig. 5. It consists of the address generator, radix-4/2 DIF butterfly unit, scaling factors generator, data RAM, and twiddle factor ROM. The data RAM employs four 2048-word two-port SRAM modules to support up to 8192-point FFT processing. Four scaling factors are produced by the scaling factors generator according to the outputs of butterfly unit. Data are scaled by right shifters in the data RAM [1].

TABLE I
Scheduling of the Twiddle Factors

k	1	2	3	4	5	6	7	8
W_{64}^{k}	W_{64}^{1}	W_{64}^{2}	W_{64}^{3}	W_{64}^{4}	W_{64}^{5}	W_{64}^{6}	W_{64}^{7}	W_{64}^{8}
W_{64}^{2k}	W_{64}^{2}	W_{64}^{4}	W_{64}^{6}	W_{64}^{8}	W_{64}^{6}	W_{64}^{4}	W_{64}^{2}	W_{64}^{0}
W_{64}^{3k}	W_{64}^{3}	W_{64}^{6}	W_{64}^{7}	W_{64}^{4}	W_{64}^{1}	W_{64}^{2}	W_{64}^{5}	W_{64}^{8}

B. Tailored Constant Multiplier Array

Constant Multipliers, composed of several adders and shifters, are widely used to reduce the datapath power dissipation [14]–[16]. Generally, they are introduced either in small transform size parallel FFT processor [14] or in pipeline architecture which can decompose a long transform size into N-dimensional small FFTs [15] [16]. Unlike those previous works, the proposed tailored constant multiplier array can only support a part of twiddle factor multiplications during the FFT calculation. Thus, both constant multipliers and three complex multipliers are implemented in the radix-4 butterfly unit. With an index determination mechanism, the FFT processor is able to decide which ones to be used during the computation. The power consumption of butterfly unit can save about 20% compared to pure complex multipliers based butterfly unit.

The twiddle factors of the proposed tailored constant multiplier array are W_{64}^{k}, where $k \in [1, 8]$. By using the symmetry feature in the complex plane [15], these constant multipliers are enough to support W_{64}^{p}-related multiplication, where p is from 0 to 63. According to Fig. 1, three twiddle factors, namely W_{64}^{k}, W_{64}^{2k}, W_{64}^{3k}, are needed simultaneously in the butterfly unit. Table I shows the scheduling of these three twiddle factors in the tailored constant multiplier array after using the mapping table mentioned in [15]. Obviously, they have different values from each other except for $k = 4$ and 8. Therefore, additional constant multiplier 4 and 8 are employed in the proposed FFT processer. Fig. 6 illustrates the architecture of the tailored constant multiplier array. It can

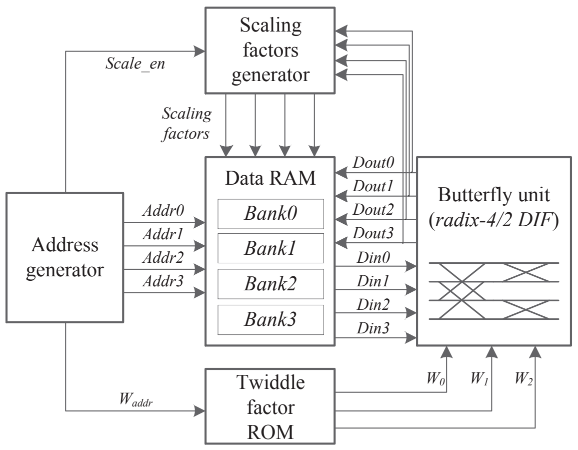

Fig. 5. Overall architecture of proposed FFT processor.

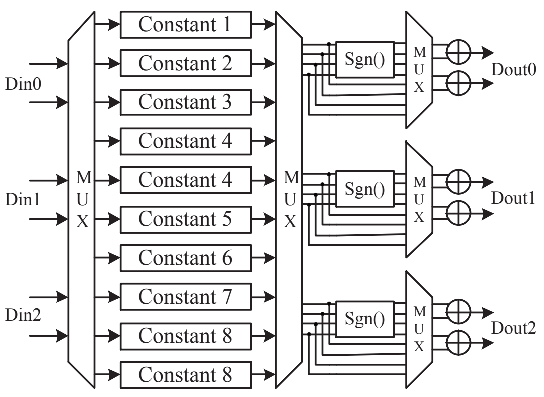

Fig. 6. The constant multiplier array architecture.

compute three complex multiplications at the same time using ten constant multipliers following with swapping the real and imaginary parts and choosing the appropriate sign.

During FFT calculation, the index determination mechanism can detect whether the twiddle factors fall within the range of W_{64}^{k} according to the index provided by address generator. If they do, then the tailored constant multiplier array is used. Otherwise, three complex multipliers are activated to perform nontrivial multiplication.

IV. Results and Comparison

The proposed FFT processor is designed in Verilog HDL and synthesized by Synopsys's Design Compiler, using the standard SMIC 130nm technology. The proposed design can perform 64 - 8192-point FFT processing on 2×11-bit complex data with a core area of 2.29 mm^2. The power consumption is 67.9 mW at 100MHz, obtained by power simulation tool Primetime PX. Also, the SQNR of 55.4 dB and 33.3 dB are achieved for 64-point and 8192-point FFT respectively.

A comparison of performance with previous works is listed in Table II. "Normalized energy per FFT point" and "normalized area" [17] are introduced to reflect the energy and area efficiency. Obviously, our design is better than [18] [19] in terms of normalized energy and area. Although our design occupies a little larger area than [20], the energy per FFT point is about 2.7 times lower than it. As for [17], it achieves better energy efficiency than our design while its core area is much large than ours.

V. Conclusions

In this paper, a hardware-efficient, low-power variable-length FFT design is proposed. A modified data scaling method and the trounding strategy are employed to improve the SQNR performance. Meanwhile, a tailored constant multiplier array is implemented to reduce the power consumption of datapath. The proposed processor can perform 64 - 8192-point FFT processing and occupies an area of 2.29 mm^2. The power consumption is 67.9 mW at 100MHz. Besides, the SQNR of 55.4 dB and 33.3 dB are achieved for 64-point and 8192-point FFT respectively.

TABLE II
Comparison of Performance for Variable-length FFT Design

	[18]	[19]	[20]	[17]	Proposed
Technology	250nm	350nm	180nm	180nm	130nm
Voltage	2.5V	3.3V	1.8V	1.8V	1.2V
FFT Size	8-4096	64-2048	2K/4K/8K	128-1024	64-8182
Frequency	200MHz	60MHz	79MHz @8K-point	51MHz	100MHz
Area	11.42mm^2	6.67mm^2	3.56mm^2	1.47mm^2	2.29mm^2
Power	400mW @1K-point	574mW	67mW @8K-point	33.3mW	67.9mW
Normalized Energy[1]	2.38nJ	8.30nJ	3.26nJ @8K-point	0.84nJ @1K-point	1.19nJ
Normalized Area[2]$\times10^3$	3.86	0.42	0.23	0.75	0.28

[1] Normalized Energy per FFT point = $\frac{Power\times Execution\ Time}{FFT\ size\times(Voltage/1.2V)^2}$
[2] Normalized Area = $\frac{Area}{FFT\ size\times(Technology/130)^2}$

References

[1] N. J. Ickes, "A micropower DSP for sensor applications," Ph.D. dissertation, Massachusetts Institute of Technology, 2008.

[2] W. Karlen, C. Mattiussi, and D. Floreano, "Sleep and wake classification with ECG and respiratory effort signals," *IEEE Trans. Biomed. Circuits Syst.*, vol. 3, no. 2, pp. 71 –78, 2009.

[3] S.-Y. Tseng and W.-C. Fang, "An effective heart rate variability processor design based on time-frequency analysis algorithm using windowed lomb periodogram," in *Proc. IEEE BioCAS*, Nov. 2010, pp. 82 –85.

[4] Z. Nie, L. Wang, W. Chen, T. Zhang, and Y. Zhang, "A low power biomedical signal processor ASIC based on hardware software codesign," in *Proc. IEEE EMBS*, 2009, pp. 2559 –2562.

[5] Y. Chen, Y.-W. Lin, Y.-C. Tsao, and C.-Y. Lee, "A 2.4-Gsample/s DVFS FFT processor for MIMO OFDM communication systems," *IEEE J. Solid-State Circuits*, vol. 43, no. 5, pp. 1260 –1273, May 2008.

[6] S.-N. Tang, C.-H. Liao, and T.-Y. Chang, "An area- and energy-efficient multimode FFT processor for WPAN/WLAN/WMAN systems," *IEEE J. Solid-State Circuits*, vol. 47, no. 6, pp. 1419 –1435, 2012.

[7] Y. Chen, Y.-C. Tsao, Y.-W. Lin, C.-H. Lin, and C.-Y. Lee, "An indexed-scaling pipelined FFT processor for OFDM-based WPAN applications," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 55, no. 2, pp. 146 –150, Feb. 2008.

[8] T. Lenart and V. Owall, "A 2048 complex point fft processor using a novel data scaling approach," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS'03)*, vol. 4. IEEE, 2003, pp. IV–45.

[9] http://www.xilinx.com/univ/teaching_materials/dsp_primer/sample/lecture_notes/FPGAArithmetic_word.pdf, Aug. 2007.

[10] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of computation*, vol. 19, no. 90, pp. 297–301, 1965.

[11] W.-H. Chang and T. Nguyen, "On the fixed-point accuracy analysis of fft algorithms," *IEEE Trans. Signal Processing*, vol. 56, no. 10, pp. 4673–4682, 2008.

[12] Y.-W. Lin, H.-Y. Liu, and C.-Y. Lee, "A dynamic scaling FFT processor for DVB-T applications," *IEEE J. Solid-State Circuits*, vol. 39, no. 11, pp. 2005 – 2013, Nov. 2004.

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

[14] K. Maharatna, E. Grass, and U. Jagdhold, "A 64-point fourier transform chip for high-speed wireless lan application using ofdm," *IEEE J. Solid-State Circuits*, vol. 39, no. 3, pp. 484–493, 2004.

[15] Y.-W. Lin, H.-Y. Liu, and C.-Y. Lee, "A 1-gs/s fft/ifft processor for uwb applications," *IEEE J. Solid-State Circuits*, vol. 40, no. 8, pp. 1726–1735, 2005.

[16] S.-N. Tang, J.-W. Tsai, and T.-Y. Chang, "A 2.4-gs/s fft processor for ofdm-based wpan applications," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 6, pp. 451–455, 2010.

[17] C.-M. Chen, C.-C. Hung, and Y.-H. Huang, "An energy-efficient partial fft processor for the ofdma communication system," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 2, pp. 136–140, 2010.

[18] G. Zhong, F. Xu, and A. N. Willson Jr, "A power-scalable reconfigurable fft/ifft ic based on a multi-processor ring," *IEEE J. Solid-State Circuits*, vol. 41, no. 2, pp. 483–495, 2006.

[19] J.-C. Kuo, C.-H. Wen, C.-H. Lin, and A.-Y. Wu, "Vlsi design of a variable-length fft/ifft processor for ofdm-based communication systems," *EURASIP journal on Applied signal processing*, vol. 2003, pp. 1306–1316, 2003.

[20] Y.-J. Cho, C.-L. Yu, T.-H. Yu, C.-Z. Zhan, and A.-Y. A. Wu, "Efficient fast fourier transform processor design for dvb-h system," in *Proc. VLSI/CAD Symposium*, 2007.