Low-Cost and Low-Complexity Sliding Discrete Fourier Transform Based on a Compact Recursive Structure for Real Input Sequence

Shin-Chi Lai[2], *Wen-Ho Juang*[1], *Yi-Hsiang Juan*[1], *Ching-Hsing Luo*[1], *and Wen-Kai Tsai*[3]
[1]Department of Electrical Engineering, National Cheng-Kung University Tainan, Taiwan
[2]Department of Computer Science and Information Engineering, Nan Hua University Chiayi, Taiwan
[3]Department of electrical engineering, National Formosa University, Yunlin, Taiwan
Email: shivan0111@nhu.edu.tw

***Abstract*—This paper proposes a novel sliding discrete Fourier transform (DFT) algorithm and its architecture design to efficiently compute the information of time-frequency spectrum. Since the sliding process is adopted sample by sample, the spectral bin output data rate can be same to the input data rate. Under the conditions of M-sample real input sequence (M=256) and N-point recursive DFT computation (N=64), the proposed method has the following advantages: 1) Proposed-I requires less computational complexity than Krzysztof Duda's method does. The proposed algorithm greatly achieves 80.35% reduction in computation because the number of multiplication only takes 44,864 operations. Additionally, 54.91% of addition operations can be saved; 2) For computing each frequency bin, the complexity of Proposed-I only requires 4 real additions and 2 real multiplications after the first spectral component has been finally calculated; 3) Proposed-II utilizes three registers and re-timing scheme to effectively shorten and balance the critical path of the proposed design. Moreover, the number of coefficients can be saved by 50% compared to Krzysztof Duda's method. In FPGA implementation, the proposed design can be operated at 43.5 MHz processing rate which can easily meet the requirement of real-time application. Therefore, it would be more suitable for real-time analysis of time-frequency spectrum in the future.**

***Index Terms*—Discrete Fourier transform (DFT); Recursive DFT (RDFT); Sliding DFT (SDFT);**

I. INTRODUCTION

In many applications of digital signal processing, Discrete Fourier Transform (DFT) has been widely employed to analysis the signal power in frequency domain. To observe the variety of time-frequency spectrum in details, a sliding sinusoidal transform [1], such as sliding DFT (SDFT) [2, 3, 4], is proposed to offer enough information in a short-time domain. Due to the recursive relationship between two subsequent local transform spectra [2, 3], the recursive DFT can be combined with the sliding unit to implement a SDFT computation. As well known, Goerzel's DFT [5] is useful and powerful in certain practical frequency bins for dual-tone multi-frequency signaling (DTMF) recognition applications. Unlike fast Fourier transform (FFT) [6, 7, 8] calculations, it takes less computational workload while fewer frequency bins are required. Recently, various recursive DFTs (RDFTs) [9-14] have been well developed. Lai *et al.* [10, 11, 12] have a greater improvement in terms of computational cycle and computational complexity. To achieve high performance in computation, a hybrid architecture design of RDFT and radix-2^r FFT [14] is recently proposed for variable-transform-length DFT. On the other hand, Krzysztof Duda proposed a powerful mSDFT algorithm by using a modulated concept [4] to obtain an exactly guaranteed accurate and stable SDFT computation. However, it caused a heavy computational workload per output sample, *i.e.* 10 real multiplications and 9 real additions more than previous works [2, 3]. Based on these experiences of above works, a low-complexity and low-cost RDFT is further extended to develop a novel SDFT algorithm with a comparable accuracy to Jacobsen and Lyons' SDFT [2, 3]. Since SDFT is applied to various signal processing topics especially for ultrasonic range finder [15], ECG noise cancelers [16], and phase locking scheme [17], a real-time, low-cost and low-complexity SDFT algorithm has become an essential issue not only for software realization but also for hardware implementation.

In this work, we focus on not only reducing the computational complexity in algorithm but also saving the hardware cost in implementation. The proposed algorithm is derived in details, and then is mapped into a hardware architecture. By using some basic very-large-scale integration (VLSI) schemes, a low-cost and high-speed SDFT hardware accelerator can be efficiently accomplished.

The rest of this paper is organized as follows: Section II takes a detailed derivation in algorithm first, and then the system transfer function of the proposed SDFT formula is given. Section III demonstrates the compact architecture design of the proposed SDFT, and Section IV compares various performance metrics with other approaches. Finally, conclusions are outlined in Section V.

II. PROPOSED ALGORITHM DERIVATION OF NOVEL SLIDING DFT COMPUTATION

The N-point DFT computation with the input (x[n]) and output (X[k]) sequences is defined as follow:

$$X[k]=\sum_{n=0}^{N-1}x[n]\times W_N^{nk}, \quad \text{where } W_N^{nk}=e^{j2\pi nk/N} \text{ and } k=0 \text{ to } N-1. \tag{1}$$

The transfer function of Goertzel DFT [5] is further derived as a z-transform formula (2) by a recursive difference equation (3). It should be noticed that the recurrent loops of Goertzel DFT totally takes (N+1) times, and the input sequence should be set zero during the $(N+1)^{th}$ time slot.

$$H_G(z)=\frac{1}{1-W_N^k z^{-1}}=\frac{1-W_N^{-k}z^{-1}}{1-2\cos(2\pi k/N)z^{-1}+z^{-2}} \tag{2}$$

$$y[n]=\left(W_N^k \times y[n-1]+x[n]\right),\ where\ y[-1]=0 \qquad (3)$$

By following up Jacobsen and Lyons' approach [2], the system transfer function of sliding Goertzel DFT [5] can be therefore defined as (4). Compared with (2) and (4), the major difference shows that the numerator of $H_{SG}(z)$ is multiplied by $(1-z^{-N})$.

$$H_{SG}(z)=\frac{\left(1-W_N^{-k}\right)\left(1-z^{-N}\right)}{1-2\cos\left(2\pi k/N\right)z^{-1}+z^{-2}} \qquad (4)$$

Since the previous work [10] had a different difference equation as (5), the transfer function of Lai *et al.*'s DFT was derived as (6). Eq. (6) was therefore expressed as (7), where θ_k was set to $2\pi k/N$. To reduce the multiplication of $\cos(\theta_k)$ in implementation, the coefficients of $\cos(\theta_k)$ and $2\cos(\theta_k)$ were shared by using one real multiplier and a shifter. Different to Goertzel's DFT, the total recurrent times of Lai *et al.*'s RDFT only take N iterations due to the derivation of (5).

$$m[n]=W_N^k \times\left(m[n-1]+x[n]\right),\ where\ m[-1]=0 \qquad (5)$$

$$H_{Lai}(z)=\frac{W_N^k-z^{-1}}{1-2\cos\left(2\pi k/N\right)z^{-1}+z^{-2}} \qquad (6)$$

$$\begin{aligned} H_{Lai}(z) &= \frac{\cos(\theta_k)+j\sin(\theta_k)-z^{-1}}{1-2\cos\theta(\theta_k)z^{-1}+z^{-2}} \\ &= \frac{j\sin(\theta_k)+\left(\cos(\theta_k)-z^{-1}\right)}{1-z^{-1}\left(2\cos(\theta_k)-z^{-1}\right)} \end{aligned} \qquad (7)$$

By multiplying the factor $(1-z^{-N})$ into $H_{Lai}(z)$, the system transfer function of the proposed SDFT, *i.e.* $H_{SLai}(z)$, could be finally yielded

$$H_{SLai}(z)=\frac{\left[j\sin(\theta_k)+\left(\cos(\theta_k)-z^{-1}\right)\right]\times\left(1-z^{-N}\right)}{1-z^{-1}\left(2\cos(\theta_k)-z^{-1}\right)}. \qquad (8)$$

Eq. (8) clearly shows that only two real multiplications are required for the coefficients of $\sin(\theta_k)$ and $\cos(\theta_k)$ in the proposed SDFT. Compared with Krzysztof Duda's mSDFT [4, Fig. 4], the proposed method really has less computational complexity.

III. Architecture Design of the Proposed Novel Sliding DFT Algorithm

A. Design Concept of Kernel Processing Unit

According to (8), the architecture design of the proposed novel sliding recursive DFT algorithm can be easily mapping into Fig. 1. It can be observed that the critical path of the proposed algorithm has $(T_M + 4T_A)$, where T_M and T_A are, respectively, the time periods of the multiplier and adder. After retiming and adding three registers into Fig. 1, a higher speed SDFT design can be drawn as shown in Fig. 2. The coefficient requirement in this work as well as previous work [10] can be easily reduced by 50% due to the symmetric identities of sine and cosine. Since the input sequence is real number, the computational complexity of the proposed SDFT algorithm would totally takes 2 real multiplications and 4 real additions for the calculation of each frequency bin after the first spectral component has been finally calculated.

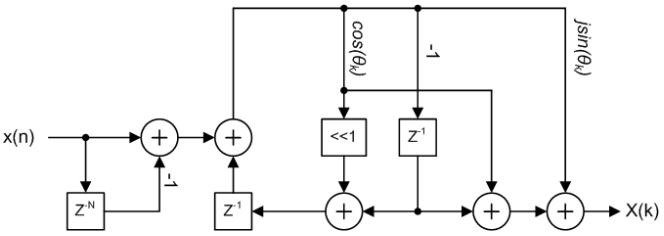

Fig. 1 Proposed Novel Sliding DFT Algorithm (Proposed I)

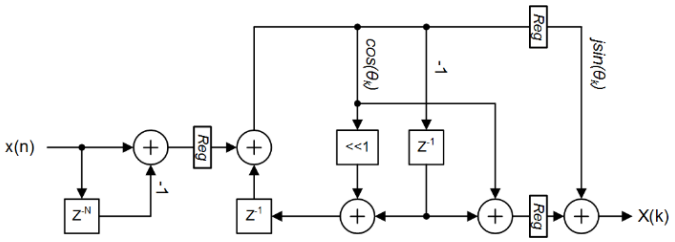

Fig. 2 Proposed Low-critical-path SDFT Design (Proposed II)

B. Overall Hardware Architecture Design

The overall hardware architecture can be mainly divided into the following three blocks: **(1)** Control Unit (CU); **(2)** Memory Unit (MU); **(3)** Processing Unit (PU). Figure 3 shows the overall architecture design, and it is noticed that the MU is composed of RAM and ROM, where ROM and RAM are, respectively, applied for storing the twiddle factors and implementing the element of Z^{-N} as mentioned in Fig. 2. The input sequence x[n] is fed to the PU for recursive computation. In general, the proposed hardware accelerator would be integrated with a greater system so that it should have a handshaking process through the valid signals of input and output, *i.e.* xValid and XValid, to make sure that the input/output data is ready. While CU receives this signal, it will follow the finite-state-machine rule to determine the action modes ('READ' or 'WRITE') of MU. At the *Initial State*, MU only executes the action of 'WRITE' until the next state, *i.e. Pre-Read and Write State*, is ready for entry. CU further controls the relative memories to finish the corresponding action modes according to the fed data for MU. While CU found that there is no input data, *i.e.* xValid is 'LOW', the coefficient ROM just can process the request of coefficient update. In the meanwhile, CU would determine that the addressing way of coefficient ROM is increased or decreased. At this moment in time, it shows that a sliding DFT operation is finished. The timing diagram of the overall architecture is displayed in Fig. 4.

C. The Control of Memory Unit

To efficiently reduce the area cost, the memory controller for the access of RAM and ROM is designed into two parts. The first one is RAM controller, and the other one is for ROM. **(1)** For RAM controller design: the SDFT computation requires an accumulated circuit in the first stage to recursively calculate the operation of $(1-Z^{-N})$, it implies that the input data should be carefully stored via the delay element of Z^{-N}. To realize this

delay element and to solve the synchronizing access of 'READ' and 'WRITE' requests, a directly mapping method instead of FIFO registers is employed by a two-port memory. However, it still costs more area and hardware resource in implementation. Therefore, we adopt two half-size and single-port memories (memory #1 and #2) with an extra smaller control circuit to achieve a two-port memory in this work. The behavior of control circuit has three states, *i.e.* "*initial*" state, the state of "*read action for memory#1 and write action for memory#2*", and the state of "*write action for memory#1 and read action for memory#2*". In the *initial* state, the action modes of these two memories are alternately writing data into to memory until (N-1) clock cycles finished. Then, memory#1 keeps writing action, and memory#2 changes the action mode into 'READ' for data pre-loading which is used under the time period of $(N+1)^{th}$ clock cycle. This state would be kept executing until the input of SDFT without any data coming. **(2)** ROM controller design: For N-point sliding window, the cosine and sine functions totally takes (2N) coefficients. In hardware implementation, we can employ the up/down counter and the symmetric identity of coefficients as mentioned above to reduce the number of coefficients by 50%. Figure 5 displays the relationships between the coefficients of sine and cosine. While cosine coefficient is under the action mode of 'READ' from memory, an up counter can be used to produce the required memory address while the index of k is increased with frequency bin changing. Then, the desired value of cosine coefficient, cos(2*pi*(k+1)/N), for next time period is further prepared, after finishing all operations during the time period of index of k. Due to the symmetric identity of cosine function, cos(2*pi*(k+1)/N) can be instead of cos(2*pi*(k-1)/N) so that the down counter could be applied to generate the desired memory address. Based on this scheme, the total size of coefficient ROM therefore takes N words.

IV. Comparison and Discussion

In this section, the performance metrics in terms of multiplication, addition, and coefficient are evaluated in Table 1 for SDFT and FFT-based SDFT algorithms. Here, we assumed that the transform length (N) of DFT is 64, the time-domain total sliding samples (M) are 256, and the input signal is real number. It should be noticed that one complex multiplication requires four real multiplications and 2 additions for the following performance evaluation. The results showed that radix-2-based FFT [8] has totally 244,992 multiplications and 244,992 additions; however, the sliding RDFT algorithms such as Jacobsen and Lyons' [2] and the proposed algorithm have quite lower computational complexity than FFT-based SDFT computation. Krzysztof Duda's [4], respectively, takes 228,352 multiplications and 207,936 additions for calculating SDFT coefficients. Compared with this latest SDFT approach, the numbers of multiplication and addition for the proposed algorithm are, respectively, 44,864 and 93,760 which have 80.35% and 54.91% reduction. Table 1 also showed that the numbers of coefficients and computational cycles of the proposed SDFT algorithm are comparable to that of [2, 4, 8]. For analyzing the execution time of hardware accelerator, the items of critical path and computational cycle are both involved. It also results that the proposed design would have similar performance but with less number of multiplication in computation. In addition, the number of coefficient for the proposed algorithm has a great reduction by 50% compared with previous works [2, 4].

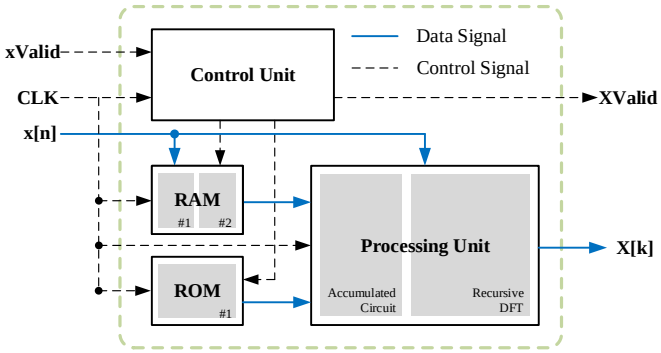

Fig. 3 Proposed Overall Hardware Architecture Design

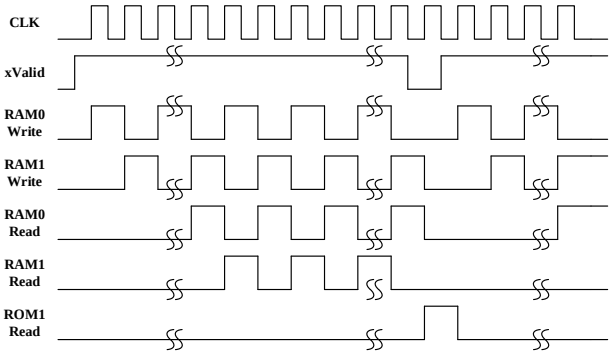

Fig. 4 Timing Diagram of the Proposed Architecture Design

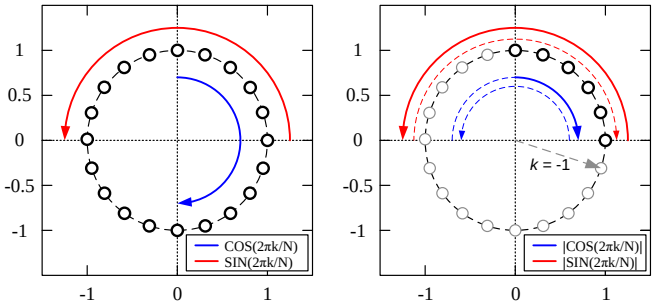

Fig. 5 Relationships between Coefficients of Sine and Cosine

Table 2 compares hardware cost in terms of multiplier, adder, multiplexer, register, and coefficient-ROM size. The result demonstrates that the proposed design has the lowest cost than other approaches. There are only 2 real multipliers, 5 real adders, 69 registers, and 64 coefficients in implementation. Compared with Krzysztof Duda's [4], the numbers of multiplier and coefficient can be, respectively, reduced by 80% and 50%. For hardware realization, the proposed design, which is implemented by Altera Cyclone IV device, can be operated at 43.5 MHz. This implies that the proposed design can produce each frequency bins within the time period of 0.0229 us. It totally takes 1.47 us for all frequency bins. Under the condition of 1 sound channel and 48 kHz sampling rate, the

real-time specific requirement is 20.833 us which is greater than that of the proposed 64-point SDFT hardware accelerator design. Overall, the proposed design would be more suitable for the real-time application of time-frequency spectrum due to its low complexity.

V. Conclusion

In this paper, a low-cost and low-complexity sliding DFT algorithm is developed and designed for the time-frequency spectra calculation. The comparison results proved that the proposed method exactly has better performance in terms of computational complexity, hardware resource and computational speed than other related approaches. It would be more powerful for future applications on the topic of digital signal processing.

Acknowledgment

This work was supported in part by the Ministry of Science and Technology, Taiwan under grant number 103-2221-E-343-001 and 104-2221-E-343-003. This work was also supported in part by NHU Research Project under grant number Y103000943. The authors would like to thank Prof. Sheau-Fang Lei and Prof. Ching-Hsing Luo for their kindly support to give abundant manpower and suggestions for this research topic.

References

[1] Vitaly Kober, "A Fast Algorithms for the Computation of Sliding Discrete Sinusoidal Transforms," IEEE Transactions on Signal Processing, vol. 52, no.6, pp. 1704–1710, June 2004.

[2] E. Jacobsen and R. Lyons, "The sliding DFT,"IEEE Signal Processing Mag., vol. 20, no. 2, pp.74–80, Mar. 2003.

[3] E. Jacobsen and R. Lyons, "An update to the sliding DFT," IEEE Signal Processing Mag., vol. 21, no.1, pp. 110–111, Jan. 2004.

[4] Krzysztof Duda, "Accurate, Guaranteed Stable, Sliding Discrete Fourier Transform," IEEE Signal Processing Mag., vol. 27, no.6, pp. 124–127, Nov. 2010.

[5] G. Goertzel, "An algorithm for the evaluation of finite trigonometric series," Amer. Math. Mon., vol. 65, pp. 34–35, Jan. 1958.

[6] Koushik Maharatna, Eckhard Grass, and Ulrich Jagdhold, "A 64-Point fourier transform chip for high-speed wireless LAN application using OFDM," IEEE Journal of Solid-State Circuits, vol. 39, no. 3, pp. 484-493, Mar. 2004.

[7] C. Yu, M.H.Yen, P.A. Hsiung, et al, "A Low-Power 64-point Pipeline FFT/IFFT Processor for OFDM Applications," IEEE Trans. Consum. Electron., vol. 57, no.1, pp.40-45, 2011.

[8] V. Arunachalam, and Alex Noel Joseph Raj, "Efficient VLSI implementation of FFT for orthogonal frequency division multiplexing applications," IET Circuits Devices Syst.,, vol. 8, no. 6, pp. 526–531, Jun. 2014.

[9] 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 Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. E90-A, no. 8, pp. 1644-1652, Aug. 2007.

[10] S. C. Lai, S. F. Lei, C. L. Chang, C. C. Lin and C. H. Luo, "Low Computational Complexity, low Power, and Low Area Design for the Implementation of Recursive DFT and IDFT Algorithms," IEEE Trans. Circuits Sys. II, vol. 56, no. 12, pp. 921-925, Dec. 2009.

[11] S. C. Lai, W. H. Juang, C. C. Lin, C. H. Luo, and S. F. Lei, "High-Throughput, Power-Efficient, Coefficient-Free and Reconfigurable Green Design for Recursive DFT in a Portable DRM Receiver," International Journal of Electrical Engineering, vol. 18, no.3, pp. 137–145, June 2011.

[12] S. C. Lai, Y. S. Lee, and S. F. Lei, "Low-Power and Optimized VLSI Implementation of Compact RDFT Processor for the Computations of DFT and IMDCT in a DRM and DRM+ Receiver," J. Low Power Electron. Appl., vol. 3, no. 2, pp. 99–113, May 2013.

[13] S. C. Lai, W.-H. Juang, Y. S. Lee, and S. F. Lei, "High-Performance RDFT Design for Applications of Digital Radio Mondiale," IEEE International Symposium on Circuits and Systems (ISCAS), pp.2601–2604, Beijing China, May 19-23, 2013.

[14] S. C. Lai, W. H. Juang, Y. S. Lee, S. H. Chen, K. H. Chen, C. C. Tsai, and C. H. Lee, "Hybrid Architecture Design for Calculating Variable-Length Fourier Transform," IEEE Transactions on Circuits and Systems-II: Express Briefs, in press, Sept. 20 2015.

[15] P. Sumathi, and P.A. Janakiraman, " SDFT-Based Ultrasonic Range Finder Using AM Continuous Wave and Online Parameter Estimation," IEEE Transactions on Instrumentation and Measurement, vol. 59, no. 8, pp. 1994–2004, August 2010.

[16] M.Z.U. Rahman, R.A. Shaik, and D.V.R.K. Reddy, "Efficient and Simplified Adaptive Noise Cancelers for ECG Sensor Based Remote Health Monitoring," IEEE Sensors Journal, vol. 12, no. 3, pp. 566–573, March 2012.

[17] S. Mishra, D. Das, R. Kumar, and P. Sumathi, "A Power-Line Interference Canceler Based on Sliding DFT Phase Locking Scheme for ECG Signals," IEEE Transactions on Instrumentation and Measurement, vol. 60, no. 1, pp. 132–142, January 2015.

Table 1. Computational Complexity Comparison for Various N-point DFT and FFT Algorithm Applied for Sliding Application

Metrics	2003 [2]	2010 [4]	2014 [8]	Proposed I & II
Multiplication	N*(3M+4N-3)	N*(10M+16N-16)	(M+N-1)*4*N/2*$\log_2$N	N*(2M+3N-4)
Addition	N*(4M+7N-4)	N*(9M+15N-15)	(M+N-1)*2*N*$\log_2$N	N*(4M+7N-7)
Coefficient	2N	2N	N	N
Cpt. Cycle (a)	N*(N+1)+M-1	N^2+M-1	Unlisted	N^2+M-1
Cri. Path (b)	$2T_A+T_M$	$2T_A+2T_M$	Unlisted	$4T_A+T_M^{(I)}$ $2T_A+T_M^{(II)}$
Cpt. Time	(a)*(b)	(a)*(b)	Unlisted	(a)*(b)

Note: Cri. Path = Critical Path; Cpt. Cycle = Computational Cycle.

Table 2. Hardware Cost Comparison for Various 64-point DFT and FFT Architecture Designs Applied for Sliding Application

Metrics	2003 [2]	2010 [4]	2014 [8]	Proposed I	Proposed II
Multiplier	5	10	768	2	2
Adder	4	7	Unlisted	5	5
Register	64+2	64+4	Unlisted	64+2	64+5
Coefficient	128	128	64	64	64