Zhao *et al.*
EURASIP Journal on Advances in Signal Processing (2022) 2022:23
https://doi.org/10.1186/s13634-022-00855-6

EURASIP Journal on Advances in Signal Processing

RESEARCH Open Access

High performance and resource efficient FFT processor based on CORDIC algorithm

Yupu Zhao, Hong Lv*, Jun Li and Lulu Zhu

*Correspondence:
yanjiusheng176@163.com
Anhui Jianzhu University, Hefei, China

Abstract

Fast Fourier Transform is widely used in communication and signal processing. I propose an improved multipath delay commutator pipelining architecture based on the radix-2 time decimation algorithm. By optimizing the intermediate data processing process and the first stage of pipelining, the architecture improves the system's computing speed and reduces the use of registers. I propose a multiplication scheme based on CORDIC and binary decomposition coding to realize complex number multiplication and constant multiplication and to eliminate the use of a multiplier. Experimental results suggest that proposed implementation has less latency and hardware utilization as compared to recently proposed implementations.

Keywords: Coordinate rotation digital computer (CORDIC), Multipath delayed commuting (MDC), Pipelined, Fast Fourier transform (FFT)

1 Introduction

Discrete Fourier Transform (DFT) plays a vital role digital signal processing [1]. Fast Fourier Transform (FFT) is a fast algorithm of DFT, which is widely used in communication [2, 3], image processing [4, 5], composite material science [6], signal processing [7–9], and other fields. DFT has a very high computational complexity ($\boldsymbol{O}(\boldsymbol{N}^2)$), so it is difficult to implement on Very Large-Scale Integration (VLSI). FFT is an efficient algorithm for calculating N-point DFT [10], which can be implemented in VLSI and is widely used in engineering practice. In the past few decades, FFT implementation has been a research hotspot in the field of information technology [11]. To meet the high performance and real-time requirements of modern applications, hardware designers have been trying to implement an efficient architecture for FFT calculation [12]. The goal of hardware designers is to provide efficient implementations primarily for high performance, low hardware resource utilization (adder, rotator, and memory), high precision, or low power consumption [13].

Most of the existing FFT processor architectures are memory-free pipelined architectures [14–20] and memory-based pipelined architectures [21–25]. The pipelined architecture enables all phases of the FFT processor to be executed in parallel, thus reducing execution time and latency, making it more suitable for real-time applications. Literature [14] put forward a kind of applicable to zero-fill area-efficient fast Fourier transform

(FFT) processor, through the use of the input data is the first phase zero filling sequence and FFT operation characteristics of the rotating factor value are 1, the FFT processor can effectively reduce the number of delay elements required, reduced the registers using the number, However, it does not improve the butterfly operation, consumes a lot of multiplier resources and increases the hardware complexity. Literature [15] proposed an improved parallel dual-path delay converter architecture and proposed a multiplication scheme based on the combination of the expanded coordinate rotation digital computer (CORDIC) algorithm and the binary expression based on the standard signed number (CSDBE) to realize the butterfly operation so that there is no multiplier used in the whole system. Its design improves the execution speed and throughput, but it does not improve the data transmission process and consumes a lot of register resources. FFT processor based on memory pipeline architecture uses RAM to store intermediate data in the process of data operation, which consumes a lot of storage resources, increases the hardware area and delay, so it cannot meet the requirements of modern engineering applications.

Through analysis, the major disadvantage of the existing design is that complex multiplication is usually implemented by embedded Digital Signal Process (DSP) block and a large number of delay elements are needed in the architecture. All these make FFT processors occupy more hardware area on FPGA and increase latency [26]. This paper proposes an improved multipath delay commutation (MDC) architecture based on the radix-2 decimation-in-time (R2DIT) FFT algorithm. The proposed architecture does not require complex multiplication and does not require on-chip memory to store the rotation factor, which reduces resource consumption and improves the processor performance. To eliminate the use of a multiplier in the system, a multiplication scheme based on CORDIC and binary decomposition coding (BDC) is proposed to realize complex multiplication and constant multiplication. The proposed multiplication scheme does not need to store the calculated rotation factor, but only stores the rotation factor Angle used for calculation, which reduces the storage requirements of memory, saves the area on the chip, and significantly improves the computing speed. The main contribution of this work is to propose the improved MDC architecture based on 2 DIT, double input and double output of data, and use the multiplication scheme based on CORDIC and BDM to realize complex multiplication and constant multiplication, replacing the multiplier in the system, to improve the resource utilization and performance. These reasonable architectural improvements eliminate the need for embedded dedicated functional blocks (DSP, BRAM). In addition, using shift registers instead of slower memory blocks to store data at each pipelining stage and reducing the use of shift registers through improvements to the MDC architecture can increase speed and reduce latency. The proposed design achieves better results in terms of speed, latency, throughput, and resource utilization.

2 Introduction to the basic principle of the algorithm

2.1 Base 2 DIT FFT algorithm

The Fast Fourier Transform (FFT) is a fast algorithm of the Discrete Fourier Transform (DFT). DFT is a method to analyze and process signals and systems. It realizes

the analysis of signals and systems from the frequency domain, which analyzes complex signals and systems more conveniently and intuitively.

According to the definition, the discrete Fourier transform (DFT) of a sequence $x(n)$ of length N can be expressed as:

$$X(k) = \mathrm{DFT}[x(n)] = \sum_{n=0}^{N-1} x(n) W_N^{kn} 0 \le k \le N-1 \tag{1}$$

In the formula $W_N^{kn} = e^{-\frac{j2\pi nk}{N}}$.

From Eq. (1), it can be seen that the direct calculation of $X(k)$ involves N complex multiplication and n-1 complex addition. The N complex multiplication can be decomposed into $4N$ real multiplication and $2N$ real addition, and the $N-1$ complex addition can be decomposed into 2 $(N-1)$ real addition. Therefore, every $X(k)$ value evaluated requires $4N$ real multiplications and $4N$ minus 2 real addition. Thus, to compute a DFT of length N requires N^2 complex multiplication and $N^2 - N$ complex addition, corresponding to $4N^2$ real multiplication and $4N^2 - 2N$ real addition. When N is very large, the computation amount is very large and it is difficult to realize in hardware. Therefore, it is necessary to improve the calculation method of DFT to reduce the computation amount of DFT.

By using the inherent characteristics of rotation factors, such as symmetry, periodicity, and reducibility, the computation of DFT can be greatly reduced. By using these characteristics, a long sequence of DFT can be decomposed into a short sequence of DFT, and the sequence can be decomposed into shorter sub-sequences according to the odd and even time sequence. Known as the time domain extraction method FFT algorithm (base -2 DIT-FFT algorithm), also known as Cooley-Tukey algorithm.

The calculation formula of the base-2 DIT-FFT algorithm is directly given here without process derivation, as shown in Equation (2) :

$$\begin{cases} X(K) = X_1(K) + W_N^k X_2(K) \\ X\left(K + \frac{N}{2}\right) = X_1(K) - W_N^k X_2(K) \end{cases} \qquad n \in \left[0, \frac{N}{2}\right],\ k \in \left[0, \frac{N}{2}\right] \tag{2}$$

In the formula

$$\begin{cases} X_1(K) = \sum\limits_{n=0}^{\left(\frac{N}{2}\right)-1} x(2n) W_{\frac{N}{2}}^{nk} \\ X_2(K) = \sum\limits_{n=0}^{\left(\frac{N}{2}\right)-1} x(2n+1) W_{N/2}^{nk} \end{cases} \qquad n \in \left[0, \frac{N}{2}\right], k \in \left[0, \frac{N}{2}\right] \tag{3}$$

Here, $X_1(K)$ and $X_2(K)$ represent the $N/2$-point DFT of even and odd sequences of $x(n)$, respectively. Formula (3) describes the base 2 butterfly operation in the DIT algorithm, as shown in Fig. 1. FFT algorithm is implemented by iterating the butterfly operation, and its corresponding implementation circuit is shown in Fig. 2. This operation requires one addition and one subtraction, followed by multiplication of complex rotation factors. Base 2 DIT-FFT is mainly to decompose n-point sequence $x(n)$ into an even sequence and an odd sequence, and then perform odd–even decomposition of the dual sequence and odd sequence, respectively, and repeat the operation until it cannot be

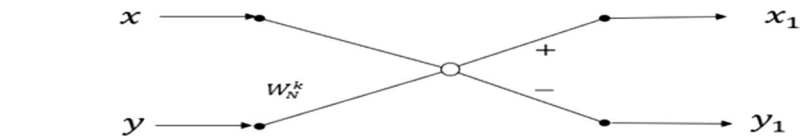

Fig. 1 Base 2 DIT butterfly operation

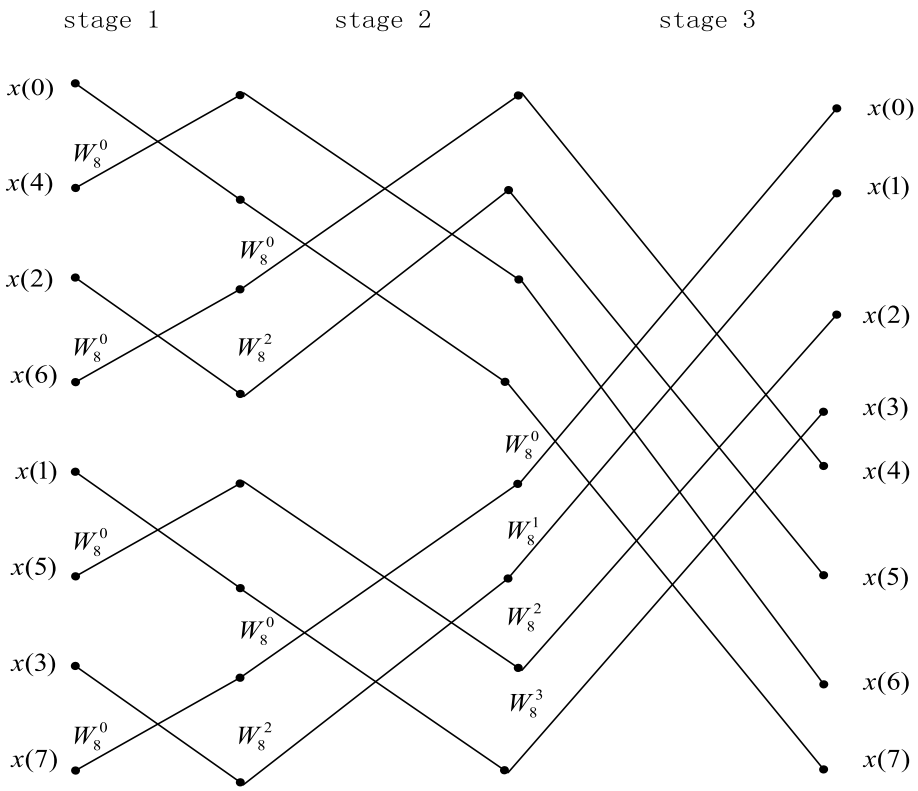

Fig. 2 8-point base 2 DIT-FFT data flow graph

decomposed. Where N is an integer power of 2. The corresponding 8-point base 2 DIT-FFT iteration process is shown in Fig. 1. The 8-point base 2 DIT-FFT data flow diagram is shown in Fig. 2, which has 3 stages and a total of 16 butterfly cells. Note that the input and output are a bit reversed, with the input in reverse order and the output in positive order.

2.2 Complex multiplication based on CORDIC algorithm

The CORDIC algorithm is widely used and is extended in literature [27] to compute a set of arithmetic functions, including multiplication, division, sine, cosine, arctangent, and hyperbolic functions. Based on the CORDIC algorithm, this paper uses simple shift, addition, and subtraction operations to achieve complex multiplication. It does not require the use of dedicated multipliers or embedded functional blocks. The use of CORDIC in the proposed architecture eliminates the complex multiplier and memory blocks needed to store the rotation factor values. The advanced complex

multiplication scheme based on CORDIC enables the proposed FFT processor to improve its processing speed and save a lot of hardware resources.

The key operation of the FFT algorithm is multiplying the operand by the rotation factor $W_N^k x(n)$, which essentially rotates $x(n)$ in the complex plane, where $\theta = \frac{2\pi k}{N}$. As shown in Fig. 3, when the coordinate is the vector rotation Angle θ of (x_i, y_i), its new coordinate can be expressed as:

$$\begin{aligned} x_{i+1} &= x_i \cos\theta - y_i \sin\theta \\ y_{i+1} &= y_i \cos\theta + x_i \sin\theta \end{aligned} \tag{4}$$

Extract $\cos\theta$, Eq. (4) can be expressed as:

$$\begin{aligned} x_{i+1} &= \cos\theta(x_i - y_i \tan\theta) \\ y_{i+1} &= \cos\theta(x_i + y_i \tan\theta) \end{aligned} \tag{5}$$

From Eq. (5), it can be seen that $\cos\theta$ only changes the magnitude of the vector. If $\cos\theta$ is removed, this rotation is called pseudo rotation, as shown in Fig. 3b. As shown in Fig. 3c, for pseudo rotation, rotation Angle θ can be decomposed into a series of a small sum of angles.

$$\theta = \sum_{i=0}^{\infty} \theta_i \tag{6}$$

We can know by using the property of tangent function $\tan\theta = d_i 2^{-i}$, in there $d_i \in \{-1, 1\}$.

The introduction of the variable z, $z_0 = \theta$, defines

$$z_{i+1} = z_i - d_i \arctan\left(2^{-i}\right) \tag{7}$$

Thus, the rotation mode iteration process of the CORDIC algorithm can be expressed as

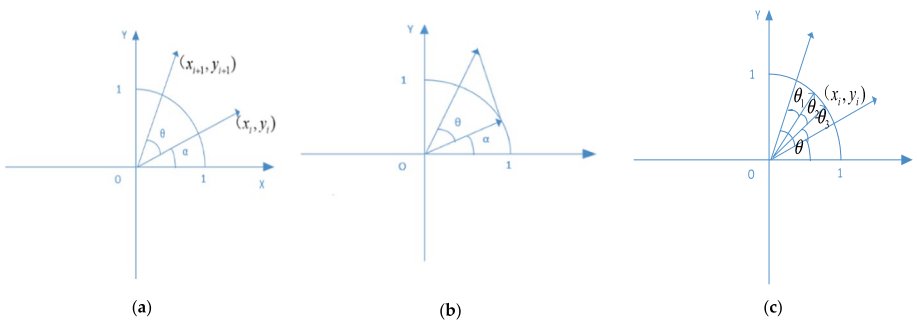

Fig. 3 Rotation mode of CORDIC algorithm in a circular system. **a** Vector rotation; **b** vector pseudo rotation; **c** it breaks down into smaller angles

$$\begin{aligned} x_{i+1} &= K\left(x_i - d_i y_i 2^{-i}\right) \\ y_{i+1} &= K\left(x_i + d_i y_i 2^{-i}\right) \\ z_{i+1} &= z_i - d_i \theta_i \\ d_i &\begin{cases} +1 & z_i \geq 0 \\ -1 & z_i < 0 \end{cases} \end{aligned} \tag{8}$$

In the formula, $K = \prod_{i=0}^{n-1} \frac{1}{\sqrt{1+2^{-2i}}}$ is the Mold length compensation factor. When the number of iterations is large enough $K \approx 0.60725$. From Eq. (8), it can be seen that complex multiplication can be achieved by shifting, adding, and subtracting.

3 Hardware implementation of FFT processor

The key to hardware implementation of the FFT algorithm is the realization of complex multiplication and architecture. In this paper, a multiplication scheme based on CORDIC and BDC is proposed to realize complex multiplication and constant multiplication, eliminating the use of a multiplier in the system. The traditional MDC architecture is optimized reasonably and an improved MDC architecture is proposed. The proposed architecture reduces the use of delay units and improves the throughput and speed of computation.

3.1 Hardware realization of complex multiplication based on CORDIC algorithm

In this study, the CORDIC algorithm is used to implement complex multiplication through the simple shift, addition, and subtraction operations. CORDIC algorithm is implemented using a parallel pipeline structure, and its implementation block diagram is shown in Figure 4. It can be seen from Figure 4 that the whole CORDIC algorithm is implemented using a 16-level pipeline, where the CORDIC iteration unit carries out iterative calculation according to the CORDIC iteration algorithm rules, and the pipeline structure at all levels is the same except for different right shift and rotation Angle. The logical structure diagram of the CORDIC iteration unit is shown in Figure 5. Figure 5 shows that the CORDIC iteration unit consists of a shifter, an adder, and a subtracter.

At each clock cycle, the controller determines the right shift and the type of operation based on the Z_i symbol bit of the Angle accumulator value. The initial Angle Z_0 is equal to the desired vector rotation Angle, and the Angle accumulator value approaches zero after 16 iterations. The accuracy of the algorithm can be improved by the number of iterations. In general, N CORDIC iterations are needed to achieve n-bit output accuracy.

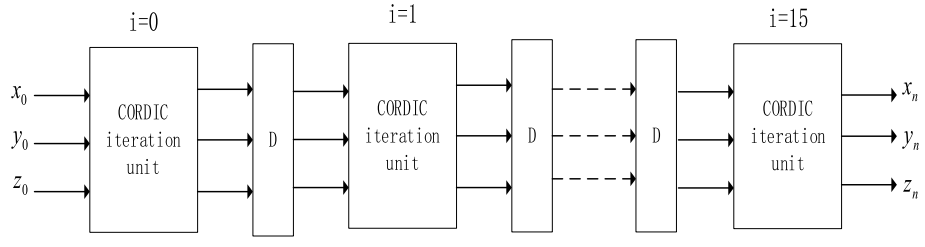

Fig. 4 Block diagram of complex multiplication based on CORDIC

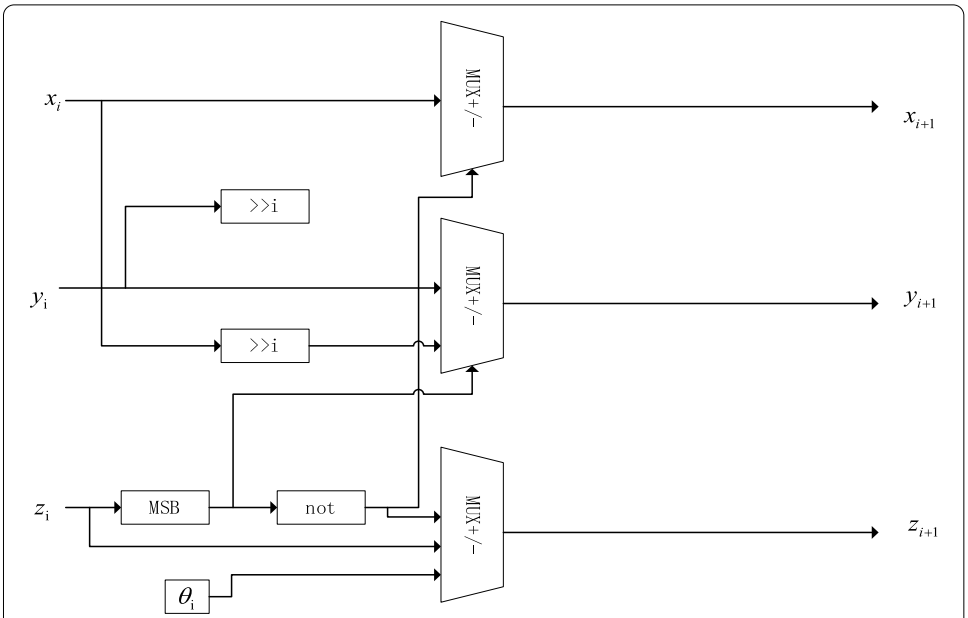

Fig. 5 CORDIC iteration module hardware architecture

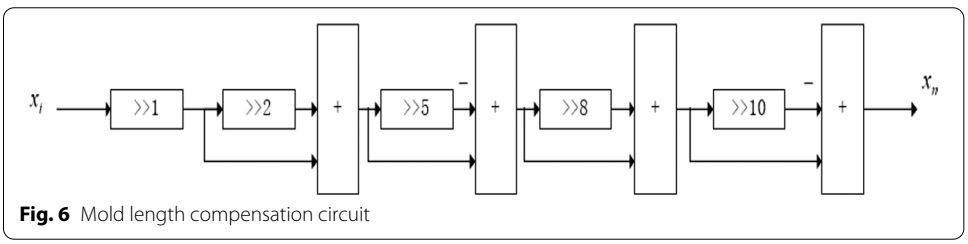

Fig. 6 Mold length compensation circuit

In this design, 16 iterations are used to implement the CORDIC algorithm, and the modulus length compensation factor K≈0.60725 is constant. The output result of the CORDIC operation module is multiplied by the modulus length compensation factor to be the final calculation result. The module length compensation factor is decomposed by BDC, and the result is shown in Eq. (9). Therefore, constant multiplication can be realized through the shift addition circuit, which improves the operation speed. The realization circuit is shown in Fig. 6.

$$K = \frac{1}{2}\left(1+2^{-2}\right)\left(1-2^{-5}\right)\left(1+2^{-8}\right)\left(1-2^{-10}\right) \tag{9}$$

3.2 Improved hardware implementation of the MDC architecture

The dual-channel delay commutator (DDC) pipeline architecture is proposed in this paper, which is a reasonable improvement on the traditional MDC architecture [28]. The traditional MDC architecture is shown in Figure 7. It can be seen from Figure 7 that the traditional MDC architecture has only one data input, and the butterfly operation unit is idle for half of the clock cycle from the first stage of operation. The

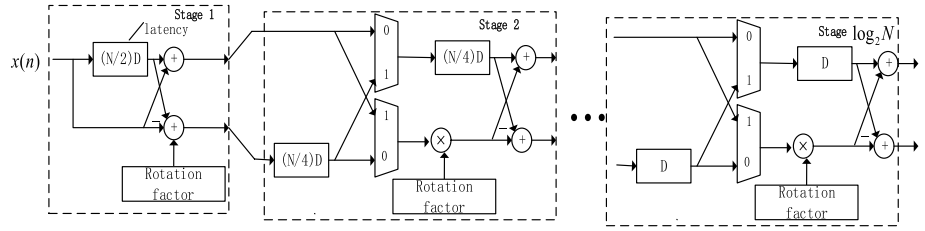

Fig. 7 Traditional MDC architecture

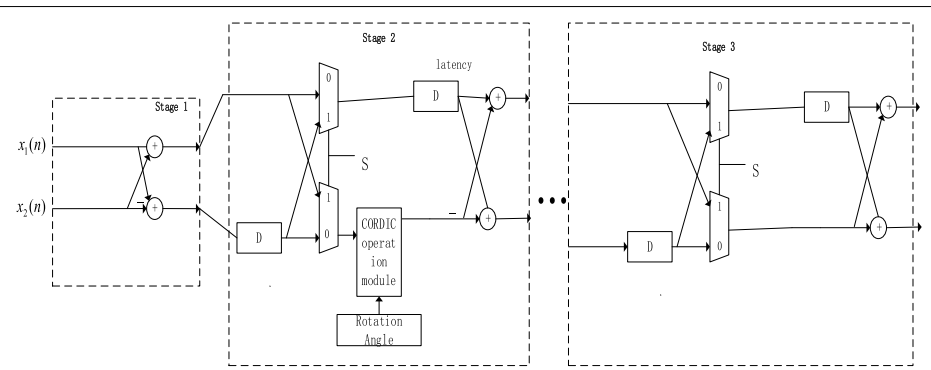

Fig. 8 8 points FFT proposed DDC architecture

working frequency of the whole system is consistent with the input data sampling rate so that the system throughput rate and system working speed is very low. Moreover, it uses RAM to store rotation factors and multipliers to realize complex multiplication, which consumes a lot of hardware resources. Therefore, the architecture can be improved from the above two points.

In the proposed DDC architecture, the data flow is continuous, and each stage has a special butterfly operation processing unit for butterfly operation. Except for the first stage, complex number multiplication in butterfly operation in each stage is realized by the CORDIC algorithm. The sample of input is divided into two parallel inputs by the shift register, which is used to store intermediate data and provide caching, with sufficient delay between the two inputs to adjust the data. The proposed architecture is shown in Fig. 8. In the first phase of the architecture, since the rotation factor is 1, the input can be directly added and subtracted according to the DIT FFT algorithm. In the subsequent stage, the shift register is used to delay the data alignment, and the commutator is used to adjust the data. The commutator consists of two data selectors, which are controlled by the control signal S. When the control signal level is high, the input data are exchanged, and when the control signal level is low, the data are transmitted as-is. And in the last phase, because there are only two rotation factors $W_N^0 = 1$ and $W_N^{\frac{N}{2}} = -J$, the last stage does not need to do complex number multiplication, you can use the adder and subtracter to achieve the butterfly operation. The data input and commutator require a delay to rearrange the data, so the last stage only has a delay of two clocks.

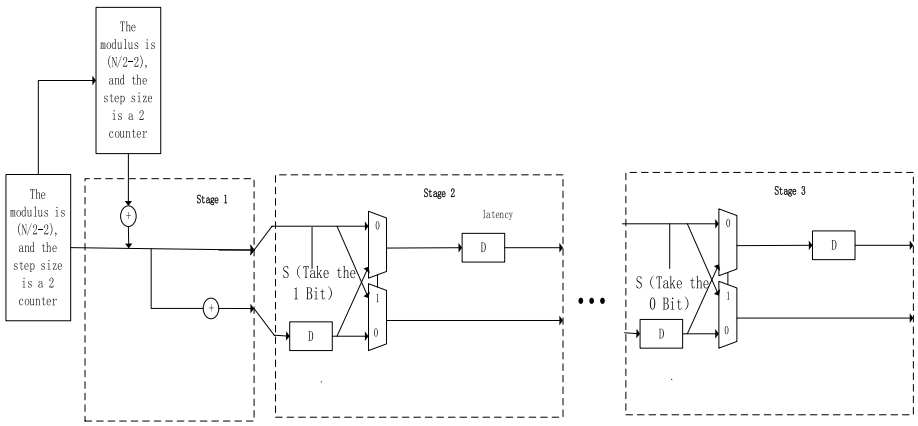

Fig. 9 8 points FFT Control signal generation circuit

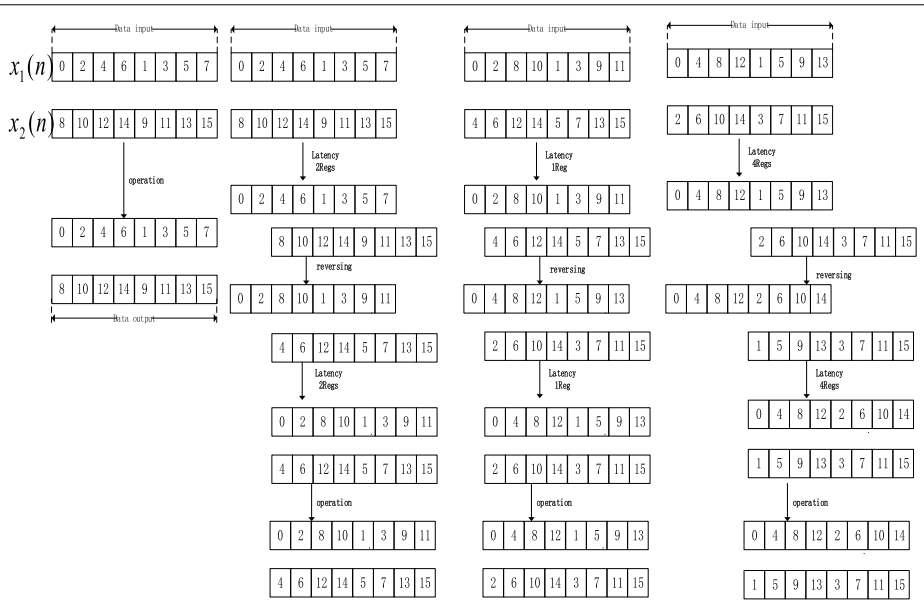

Fig. 10 16 points to improve the MDC architecture data flow diagram

The key of design is how to generate control signals at all levels. This paper adopts counter, delay unit and commutator to constitute the control signal generation unit. Figure 9 shows the 8-point FFT control signal generation unit. Figure 9 shows that the first-level control signal is equal to the first bit of the input operand address, and the last level control signal is equal to the 0th bit of the input address. This shows that for FFT processor with M stages pipeline, the value of the control signal in the p stage is equal to the value of the m-P bit of the input operand address.

The 16-point data flow diagram of the improved MDC architecture is shown in Fig. 10. In the data flow diagram, the number represents the address of the data for ease of illustration. According to the data flow diagram, the data of the improved MDC architecture is input through two channels, $x_1(n)$ and $x_2(n)$. In the first level operation unit, the data can be operated directly without delay and transform direction. In the second level

operation unit, the upper operand is delayed first, and then under the control of the reversing control signal, reverse the two inputs, and finally, the lower operand is delayed. The two inputs are now aligned, and the next step is to do the butterfly operation and output the result to the next stage. The operation process of each subsequent level is the same as that of the second level, except for the change of reversing control signal, delay depth, and rotation Angle address.

4 Analysis of experimental results

In the FFT algorithm, if the input is assumed to be real, and in the interval [−1,1), the output is in the interval [−*N*, *N*). In practice, it can be expressed as FPX (16,14). To prevent data overflow and ensure the accuracy of data, 16-bit truncation is used in the final output butterfly stage.

Verilog HDL is used to synthesize and implement the proposed FFT processor architecture. The CORDIC algorithm is used to realize the complex multiplication operation in the FFT algorithm, and the improved MDC architecture proposed in this paper is used to realize the 512 points, 1024 points, 2048 points, and 4096 points FAST Fourier transform. The input signal of FFT algorithm is: $s(t) = 0.5\cos 2\pi f_1 t + 0.5\sin 2\pi f_2 t$, $f_1 = 1000$ Hz, $f_2 = 2000$ Hz. The proposed FFT processor is evaluated in terms of execution time, logical resource consumption, and delay period. Table 1 shows a comparison between the proposed design and the traditional full-line DDC implementation of the FFT algorithm [15], in which two parameters are considered, namely the resource utilization of the slices used and the delay of the clock cycle used to calculate the input sequence FFT. In reference [15], the traditional full-pipelined DC architecture is introduced, it uses CLB to multiply complex numbers. Table 1 shows that our proposed implementation uses an average of 36.96% less LUT slicing and uses fewer resources than a traditional DDC architecture. The reduction in resources is possible because of the CORDIC algorithm for complex number multiplication, simple shift for constant multiplication through BDC, and architectural improvements that reduce the use of delay elements. As shown in Table 1, the proposed implementation has a 60.13% reduction in latency compared to traditional pipelined DC, and the proposed construction has less latency.

To further illustrate the feasibility and performance of the FFT processor architecture proposed in this paper, different clock frequencies are used to implement THE FFT

Table 1 Resource utilization and performance of the proposed architecture

FFT points	Conventional design		Proposed design		% Improvement in performance	
	Slice	Latency	Slice	Latency	Slice	Latency
512	14,844	520	9981	190	32.76	63.46
1024	20,195	1033	12,647	363	37.38	64.86
2048	24,827	2058	16,098	784	35.16	61.90
4096	34,383	4107	19,757	1671	42.54	50.31
Average performance improvement					36.96	60.13

processor. The running speed of the Proposed FFT processor architecture at each frequency is shown in Figure 11.

To illustrate the advantages of the proposed architecture, Table 2 shows a detailed comparison between our proposed pipeline architecture and other pipeline architectures, taking into account the parameters: input data width, resource utilization, latency, throughput. The length of FFT is considered a classification of the FFT architecture. The different the architecture for im-lamenting FFT algorithms are compared in Table 2. Since the order of the input and output samples depends on different applications, the hardware required to sort the data before and after FFT is not considered for comparison purposes. Some architectures use special functional blocks of FPGA, namely DSP and block RAM. For comparison, these functional blocks are converted into their equivalent pieces. A block of RAM is approximately 550 slices because a slice can be configured to store 64-bit data. A DSP block can be approximately 385 pieces because the HEIGHT of the DSP block matches a block of RAM, with each DSP block horizontally aligned to an 18K block of RAM. As you can see from Table 2, the proposed architecture is superior to the existing architecture in terms of latency, throughput, and resource utilization.

In terms of power consumption, the total power consumption of the whole FPGA design is, respectively, composed of chip static power consumption, design static power consumption, and design dynamic power consumption. The size of chip static power consumption and design static power consumption mainly depends on the level used by the chip and the occupation of logic and wiring resources in FPGA [32]. The power consumption of these two parts depends on the FPGA chip and hardware itself and has nothing to do with FFT implementation architecture. The design dynamic power consumption is related to the FFT implementation architecture and accounts for 90% of the total power consumption. The size of the designed dynamic power is related to the size of the resource consumption in FPGA, that is, when the resource consumption is the smallest, the power consumption is the lowest. Therefore, when analyzing the power consumption of the proposed FFT architecture, we mainly analyze the hardware resources consumed by FFT implementation. It can be seen from Table 2 that of all the

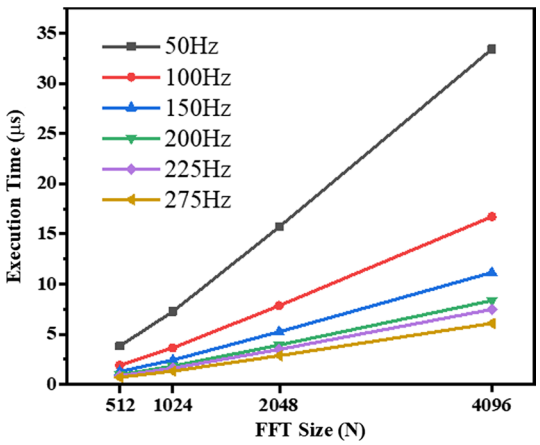

Fig. 11 The processing speed of the proposed architecture at various operating frequencies

Table 2 The proposed architecture is compared with the implemented architecture

No. of points	Design scheme	Data width	No. of slices	Block RAM	DSP	Total no. of slices	Latency (clock cycles)	Throughput (samples/cycle)	Operating frequency (MHZ)
512	Wang [31]	16-bit FXP	3681	21	5	14,516	530	2	200
	Chandra [29]	12-bit FXP	33,640	0	0	12,087	249	2	200
	Changela [30]	16-bit FXP	11,302	0	0	11,302	270	2	230
	Nguyen [15]	16-bit FXP	12,109	0	0	12,109	263	2	200
	Proposed	16-bit FXP	9981	0	0	9981	190	2	275
1024	Wang [31]	16-bit FXP	8678	48	12	34,382	722	2	200
	Chandra [29]	12-bit FXP	20,035	0	0	15,168	500	2	200
	Changela [30]	16-bit FXP	13,449	0	0	13,449	527	2	230
	Nguyen [15]	16-bit FXP	15,452	0	0	15,452	520	2	200
	Proposed	16-bit FXP	12,647	0	0	12,647	363	2	275
2048	Wang [31]	16-bit FXP	9223	54	23	43,333	1452	4	200
	Chandra [29]	12-bit FXP	24,389	0	0	19,242	1100	2	200
	Changela [30]	16-bit FXP	16,273	0	0	16,273	1040	2	200
	Nguyen [15]	16-bit FXP	19,414	0	0	19,414	1033	2	200
	Proposed	16-bit FXP	16,098	0	0	16,098	784	2	275
4096	Chandra [28]	12-bit FXP	33,640	0	0	26,343	2028	2	200
	Changela [29]	16-bit FXP	20,615	0	0	20,615	2056	2	225
	Nguyen [15]	16-bit FXP	26,543	0	0	26,543	2058	2	200
	Proposed	16-bit FXP	19,757	0	0	19,757	1671	2	275

architectures, our proposed architecture consumes the least hardware resources and therefore the lowest power consumption.

5 Conclusion

A full-flow DDC architecture is proposed, which uses the CORDIC algorithm to multiply complex numbers and simple shift and addition to multiply real numbers, with high throughput and low latency. By implementing 1024- points, 2048- points, and 4096-points fast Fourier transforms, the performance of the proposed design is evaluated in terms of throughput, latency, speed, and resource utilization, and compared with existing designs. The design effectively improves the multi-data center architecture, reduces the use of delay elements in the processor, and improves the computing

speed of the system. CORDIC algorithm is used to realize complex multiplication, which improves the design performance and resource utilization. CORDIC complex multipliers run faster and require fewer hardware resources. The design also optimizes the storage of rotation factors typically stored in block memory in traditional designs, thereby reducing the chip area and cost of the FFT processor while improving its performance due to slower memory operations. The proposed architecture is faster, simpler, more efficient, and less costly than existing architectures, providing better performance with fewer hardware resources, costs, and power consumption. Experimental results show that this algorithm achieves the highest performance in throughput, latency, speed, and resource utilization.

Acknowledgements
Not applicable.

Authors' contributions
All the work was done under the guidance of Professor Hong Lv.Conceptualization, YZ and HL .; investigation, YZ,LZ and HL; methodology, YZ,JL and HL; project administration, HL; software, YZ and LZ; supervision, HL; validation, YZ,JL and HL ; writing—original draft, YZ ,JL and HL. All authors read and approved the final manuscript.

Authors' Information
Yupu Zhao He was born in Fuyang, Anhui Province, China in 1997. He received his bachelor's degree in communication engineering from the School of Electronics and Information Engineering, Anhui Jianzhu University in 2019, and is currently pursuing a master's degree in circuits and systems at Anhui Jianzhu University. His research interests include digital signal processing and the design of FPGA architectures for digital communications.
Lv Hong, female, born in 1959 in Huining, Anhui Province, professor, master tutor, academic technology leader in Anhui Province. She has presided over and completed 7 provincial and ministerial level or above projects, such as national Natural Science Foundation of China and provincial Natural Science Foundation of China. She has published more than 60 papers in academic journals and conference proceedings at home and abroad. It has obtained eight invention patents and utility model patents, and more than a dozen software Copyrights. Her research interest covers intelligent detection, signal processing and computer applications.
Jun Li He was born in 1995. A master's degree. His research interests are in signal and information processing.
Lulu Zhu He was born in 1995. A master's degree. His research interests are in the hardware design.

Funding
This research was funded by the National Natural Science Foundation of China (No. 61372094, 61071001).

Availability of data and materials
Data sharing does not apply to this article as no datasets were generated or analyzed during the current study.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
YuPu Zhao, Hong Lv, Jun Li, and LuLu Zhu declare that they have no conflict of interests.

Received: 11 August 2021 Accepted: 7 March 2022
Published online: 21 March 2022

References

1. M.S. Kavitha, P. Rangarajan, An efficient FPGA architecture for reconfigurable FFT processor incorporating an integration of an improved CORDIC and Radix-2 r Algorithm. Circuits Syst. Signal Process. **39**, 5801–5829 (2020)
2. H.J. Lin, C.A. Shen, The architectural optimizations of a low-complexity and low-latency FFT processor for MIMO-OFDM communication systems. J. Signal Process. Syst. **93**, 67–78 (2020)
3. J.S. Bruno, V. Almenar, J. Valls, FPGA implementation of a 10 GS/s variable-length FFT for OFDM-based optical communication systems. Microprocess. Microsyst. **64**, 195–204 (2019)
4. L. Li, A.M. Wyrwicz, Parallel 2D FFT implementation on FPGA suitable for real-time MR image processing. Rev. Sci. Instrum. **89**(9), 093706 (2018)
5. C. Tian, Q. Zhang, G. Sun et al., FFT consolidated sparse and collaborative representation for image classification. Arab. J. Sci. Eng. **43**(2), 741–758 (2018)

6. S. Brisard, L. Dormieux, FFT-based methods for the mechanics of composites: A general variational framework. Comput. Mater. Sci. **49**(3), 663–671 (2010)
7. A. Chen, X. Wang, An image watermarking scheme based on DWT and DFT. In: 2017 2nd International Conference on Multimedia and Image Processing (ICMP). IEEE, 2017, pp. 177–180
8. A. Wahbi, A. Roukhe, L. Helou, Enhancing the quality of voice communications by acoustic noise cancellation (ANC) using a low-cost adaptive algorithm based Fast Fourier Transform (FFT) and circular convolution. In: 2014 9th International Conference on Intelligent Systems: Theories and Applications (SITA-14). IEEE, 2014, pp. 1–7
9. S.N. Tang, F.C. Jan, Energy-efficient and calibration-aware Fourier-domain OCT imaging processor. IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **27**(6), 1390–1403 (2019)
10. Y. Gao, Digital Signal Processing Based on FPGA. Publishing House of Electronics Industry, 2012
11. V. Kumar, D. Selvakumar, P.M. Sobha, Area, and frequency optimized 1024 point Radix-2 FFT processor on FPGA. In: 2015 International Conference on VLSI Systems, Architecture, Technology and Applications (VLSI-SATA). IEEE, 2015, pp. 1–6
12. M. Garrido, J. Grajal, M.A. Sanchez et al., Pipelined radix-2k feedforward FFT architectures. IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **21**(1), 23–32 (2011)
13. M. Garrido, M.L. López-Vallejo, S.G. Chen, Guest editorial: special section on fast Fourier transform (FFT) hardware implementations. J. Signal Process. Syst. **90**(11), 1581–1582 (2018)
14. Y. Jung, J. Cho, S. Lee et al., Area-Efficient Pipelined FFT Processor for Zero-Padded Signals. Electronics **8**(12), 1397 (2019)
15. N.H. Nguyen, S.A. Khan, C.H. Kim et al., A high-performance, resource-efficient, reconfigurable parallel-pipelined FFT processor for FPGA platforms. Microprocess. Microsyst. **60**, 96–106 (2018)
16. P.K. Godi, B.T. Krishna, P. Kotipalli, Design optimization of multiplier-free parallel pipelined FFT on field-programmable gate array. IET Circuits Devices Syst. **14**(7), 995–1000 (2020)
17. C. Ingemarsson, P. Källström, F. Qureshi et al., Efficient FPGA mapping of pipeline SDF FFT cores. IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **25**(9), 2486–2497 (2017)
18. S.L.M. Hassan, N. Sulaiman, I.S.A. Halim, Low power pipelined FFT processor architecture on FPGA. In: 2018 9th IEEE Control and System Graduate Research Colloquium (ICSGRC). IEEE, 2018, pp. 31–34
19. T. Siu, C.W. Sham, F.C.M. Lau, Operating frequency improvement on FPGA implementation of a pipeline large-FFT processor. In: 2017 19th International Conference on Advanced Communication Technology (ICACT). IEEE, 2017, pp. 5–9
20. A. Manimaran, S. Parasuraman, Design of optimized Radix-4 FFT processor with multiplier sharing method. In: IOP Conference Series: Materials Science and Engineering. IOP Publishing, 2020, **925**(1), 012067
21. Q.J. Xing, Z.G. Ma, Y.K. Xu, A novel conflict-free parallel memory access scheme for FFT processors. IEEE Trans. Circuits Syst. II Express Briefs **64**(11), 1347–1351 (2017)
22. C.F. Hsiao, Y. Chen, C.Y. Lee, A generalized mixed-radix algorithm for memory-based FFT processors. IEEE Trans. Circuits Syst. II Express Briefs **57**(1), 26 (2010)
23. A. Yazdanialahabadi, M. Ardakani, A distributed low-complexity coding solution for large-scale distributed FFT. IEEE Trans. Commun. **68**(11), 6617–6628 (2020)
24. K. Elango, K. Muniandi, VLSI implementation of an area and energy-efficient FFT/IFFT core for MIMO-OFDM applications. Ann. Telecommun. **75**, 215–227 (2019)
25. J. Wang, S. Li, X. Li, Scheduling of data access for the Radix-2k fft processor using single-port memory. IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **28**, 1676–1689 (2020)
26. S.M. Joshi, FFT architectures: a review. Int. J. Comput. Appl **116**(7), 887 (2015)
27. J.S. Walther, A unified algorithm for elementary functions. Proc. Spring Joint Comput. Conf. **38**, 379–385 (1971)
28. Y.W. Lin, H.Y. Liu, C.Y. Lee, A 1-GS/s FFT/IFFT processor for UWB applications. IEEE J. Solid-State Circuits **40**, 1726–1735 (2005)
29. D.S. Chandra Inguva, D.J.B. Seventiline, Implementation of FPGA design of FFT architecture based on CORDIC algorithm. Int. J. Electron. **108**, 1914–1939 (2021)
30. A. Changela, M. Zaveri, D. Verma, FPGA implementation of high-performance, resource-efficient Radix-16 CORDIC rotator based FFT algorithm. Integration **73**, 89–100 (2020)
31. J. Wang, C. Xiong, K. Zhang et al., A mixed-decimation MDF architecture for Radix-2k parallel FFT. IEEE Trans. Very Large Scale Integr. Syst. **24**(1), 67–78 (2015)
32. A. Drozd, S. Antoshchuk, J. Drozd, et al. Checkable FPGA design: energy consumption, throughput and trustworthiness. In: Green IT Engineering: Social, Business and Industrial Applications. Springer, Cham, 2019, pp. 73–94

Publisher's Note