remote sensing

MDPI

Article

Miniaturization Design of High-Integration Unmanned Aerial Vehicle-Borne Video Synthetic Aperture Radar Real-Time Imaging Processing Component

Tao Yang [1,*], Tong Wang [1], Nannan Zheng [1], Shuangxi Zhang [2], Fanteng Meng [1], Xinyu Zhang [1] and Qirui Wu [1]

[1] School of Aerospace Science and Technology, Xidian University, Xi'an 710126, China; wtqxy1314@stu.xidian.edu.cn (T.W.); michael.n.zheng@gmail.com (N.Z.); ftmeng@stu.xidian.edu.cn (F.M.); xinyu_z@stu.xidian.edu.cn (X.Z.); qrwu@xidian.edu.cn (Q.W.)

[2] School of Electronics and Information, Northwestern Polytechnical University, Xi'an 710129, China; zhangsx@nwpu.edu.cn

* Correspondence: taoyang@mail.xidian.edu.cn

Abstract: The unmanned aerial vehicle (UAV)-borne video synthetic aperture radar (SAR) possesses the characteristic of having high-continuous-frame-rate imaging, which is conducive to the real-time monitoring of ground-moving targets. The real-time imaging-processing system for UAV-borne video SAR (ViSAR) requires miniaturization, low power consumption, high frame rate, and high-resolution imaging. In order to achieve high-frame-rate real-time imaging on limited payload-carrying platforms, this study proposes a miniaturization design of a high-integration UAV-borne ViSAR real-time imaging-processing component (MRIPC). The proposed design integrates functions such as broadband signal generation, high-speed real-time sampling, and real-time SAR imaging processing on a single-chip FPGA. The parallel access mechanism using multiple sets of high-speed data buffers increases the data access throughput and solves the problem of data access bandwidth. The range-Doppler (RD) algorithm and map-drift (MD) algorithm are optimized using parallel multiplexing, achieving a balance between computing speed and hardware resources. The test results have verified that our proposed component is effective for the real-time processing of 2048 × 2048 single-precision floating-point data points to realize a 5 Hz imaging frame rate and 0.15 m imaging resolution, satisfying the requirements of real-time ViSAR-imaging processing.

Keywords: video synthetic aperture radar; real-time imaging; field-programmable gate array; range-Doppler algorithm; map-drift algorithm

Citation: Yang, T.; Wang, T.; Zheng, N.; Zhang, S.; Meng, F.; Zhang, X.; Wu, Q. Miniaturization Design of High-Integration Unmanned Aerial Vehicle-Borne Video Synthetic Aperture Radar Real-Time Imaging Processing Component. *Remote Sens.* **2024**, *16*, 1273. https://doi.org/10.3390/rs16071273

Academic Editors: Jianlai Chen, Yi Xiong, Hanwen Yu, Jian Peng, Mengdao Xing and Yang Lan

Received: 22 February 2024
Revised: 29 March 2024
Accepted: 2 April 2024
Published: 4 April 2024

1. Introduction

Owing to its unique characteristics, such as its full-time capabilities, being all-weather, and having a long operating range and high resolution, synthetic aperture radar (SAR) has become a new means of real-time ground detection technology. In contrast to conventional SAR systems that have a longer accumulation time using a synthetic aperture and a lower imaging frame rate [1], video SAR (ViSAR) is capable of conducting continuous detection on a ground imaging area, which can form high-continuous-frame-rate images while achieving real-time imaging detection and tracking of ground targets. For airborne ViSAR, when data are transmitted to a ground station for imaging processing, the imaging time-sensitive targets may result in failure, posing great challenges to both the flow of imaging algorithms and the platforms of imaging processing [2–4].

In the miniaturization of airborne SAR systems, significant progress has been found in existing research. In 2008, the United States successively developed micro-SAR and nano-SAR systems. The former system has a weight of 2 kg, a resolution of 1 m, and an operating range of 0.7 km. Compared to the former, the latter has improved system integration, possessing a lighter weight and longer operating range [5,6]. Based upon the

nano-SAR, in 2013, Nano SAR_ B and Nano SAR_ C emerged consecutively, which have been improved not only in resolution and operating range but also in functions that are more comprehensive and versatile in contrast to the original nano-SAR system [7]. In 2022, Wiehle et al. integrated the image formation and processing of an airborne SAR system into a multiprocessor system-on-a-chip (MPSoC) and ported the high-complexity computation of the system onto a field-programmable gate array (FPGA), and the low-complexity computation on a SAR processing system, by which the first airborne SAR system was realized on a single hardware platform [8].

The imaging algorithms of SAR can mainly be classified into two categories, i.e., time-domain algorithms and frequency-domain algorithms. Among them, the most commonly used ones in the field of ViSAR imaging processing include the time-domain back-projection (BP) algorithm, frequency-domain range-Doppler (RD) algorithm, and polar coordinate format algorithm (PFA). The BP algorithm executes a point-wise traverse to obtain precisely focused images, which requires a considerable amount of computation and thus limits its applicability [9,10]. Both the RD and PFA are fundamental and representative imaging algorithms with low computational complexity, which can be easily ported and further developed. However, the interpolation process of the PFA algorithm not only increases the computational complexity but also has a high time loss, which is inapplicable to realizing the real-time imaging processing of airborne ViSAR [11,12]. According to different domains of signal processing, the RD algorithm can be executed in the time domain or frequency domain [13]. Specifically, the frequency-domain RD algorithm uses Fresnel approximation in calculations, which can greatly minimize the computational complexity while making it more suitable for realizing high-frame-rate real-time imaging in the airborne miniaturization design of a high-integration unmanned aerial vehicle (UAV)-borne ViSAR real-time imaging processing component (MRIPC). In addition, for airborne ViSAR, the Doppler parameters of airborne radar (including the Doppler center and frequency modulation) constantly vary with time due to the changes in aircraft velocity and airflow. Therefore, motion error compensation is required for the results of RD imaging to realize high-frame-rate real-time high-quality imaging. The commonly used motion error compensation algorithms include the map-drift (MD) algorithm [14] and the phase-gradient autofocus (PGA) algorithm. Despite the fact that the PGA algorithm exhibits high accuracy and excellent stability, it needs to execute a large amount of computation and is not feasible for implementing on the FPGA. The MD algorithm is a motion error compensation algorithm based on sub-aperture processing. Compared to such algorithms, full-aperture processing can avoid the grating lobe and image-stitching problems caused by sub-aperture processing [15]. In 2023, Chen et al. proposed a parametric motion error compensation algorithm based on full-aperture processing to address the severe motion error challenges encountered in state-of-the-art airborne microwave photonic SAR systems, demonstrating superior quantitative performance compared to the alternative full-aperture motion error compensation methods [16]. Comparatively, the MD algorithm has lower computational complexity and is more suitable for application in ViSAR imaging scenarios.

The high integration of ViSAR real-time imaging processing components also puts higher demands on the computing power of hardware platforms. Currently, the research on onboard real-time processing mainly involves three commonly used platforms, i.e., digital signal processing (DSP) [17], graphic processing unit (GPU) [18–20], and FPGA [21–25]. In 2022, Wang et al. reported a terahertz SAR imaging processing system based on a multi-core DSP. The reported system can complete the processing of a 2K $\times$ 2K image in 0.755 s in range and azimuth directions and is capable of outputting five high-quality images with a 1K resolution at an imaging rate of five frames per second [17]. However, due to its serial processing mode, the DSP is inapplicable for the high-frame-rate real-time imaging of ViSAR. In 2023, our previous work on the real-time imaging processing of ViSAR was based on an embedded GPU platform [20]. However, the embedded GPU serves solely as a data processing platform, while a complete ViSAR imaging system involves numerous interfaces and entails higher complexity. Subsequently, Mota et al. proposed a high-performance

and energy-efficient hardware kernel for the configurable BP algorithm in 2023. They performed the optimized BP algorithm on the images of 512 × 512 and 1024 × 1024 pixels on FPGA, which consumed 0.14 s and 1.11 s, respectively [25], verifying the feasibility of adopting the FPGA platform for real-time ViSAR processing.

In this study, we proposed a miniaturized design of a highly integrated real-time imaging processing component for the UAV-borne ViSAR (MRIPC). This component reduces system complexity while satisfying the requirements for UAV-borne ViSAR imaging. The key contributions of our work can be formulated as follows.

Firstly, we proposed a solution for the ViSAR real-time imaging processing component based on a single large-scale FPGA chip, which integrates the functions of broadband signal generation, high-speed real-time sampling, and real-time SAR imaging processing, achieving parameterized SAR signal generation, acquisition, control, and storage.

Secondly, we adopted a concurrent access mechanism with multiple sets of high-speed data buffers to increase the data access throughput, thereby solving the problem of data access bandwidth. By flexibly allocating the DDR3 storage unit, the computation speed of the RD algorithm was improved. We also doubled the working frequency of the sub-block processing unit and divided the data block into seven sub-blocks in the azimuth direction. By reusing four-way sub-block parallel processing units, the seven-way fully parallel processing scheme was replaced, thereby balancing the computation speed of the MD algorithm and hardware resources.

Thirdly, we validated the performance of the ViSAR real-time imaging processing component using measured data. The experimental results have proven that the real-time imaging processing component of ViSAR takes only 0.2 s to process the data of 2048 × 2048 points to produce a one-frame image with a resolution of 0.15 m. The results have also validated the effectiveness of this component on 2048 × 2048 single-precision floating-point data, which can achieve real-time imaging at a high frame rate of 5 Hz, thereby verifying the reliability and effectiveness of the high-integration UAV-born ViSAR real-time imaging processing component while satisfying the requirements of ViSAR real-time imaging processing.

The remainder of this paper is organized as follows. Section 2 introduces the RD and MD algorithms. Section 3 outlines the design of our proposed MRIPC component, focusing on the methods of implementing and optimizing RD and MD algorithms at a high frame rate on a single FPGA. Section 4 validates the effectiveness of our proposed method using measured data and conducts resource consumption analysis. Section 5 summarizes the conclusions of this study.

2. ViSAR Imaging Algorithms Based on RD and MD

2.1. RD Algorithms

The rationale of the RD algorithm is to decompose two-dimensional (2D) processing into one-dimensional (1D) processing, i.e., to process in the range direction and azimuth direction separately. After preprocessing the raw data, 2D results in the frequency domain can be obtained by conducting fast Fourier transform (FFT) on the range direction and azimuth direction. In the 2D frequency domain, the above results can be converted into 2D time-domain results by multiplying them with coefficients, such as the 2D decoupling term and range pulse compression term. Then, we performed a dechirp operation in the azimuth time domain to ultimately output the imaging results in the range time and azimuth frequency domains [26,27]. The specific steps of the above operation are presented as follows.

Step 1. Perform range FFT on the preprocessed single frame of raw data collected by the component and transform the data into the range direction in the frequency domain and azimuth direction in the time domain. Use the data collected by the inertial navigation system (INS) to compensate for motion errors.

Step 2. Perform azimuth FFT on the data that are compensated with the linear phase function, then transform the obtained result into a range–azimuth frequency domain.

Step 3. Conduct quadratic range pulse compression and use the range cell migration correction (RCMC) function in the range–azimuth frequency domain to complete the range compression, quadratic range pulse compression, and range migration correction.

Step 4. Perform inverse FFT (IFFT) in the range direction and use a matching function to compensate for the result in the azimuth direction.

Step 5. Perform IFFT with respect to the above result, transforming the signal into a time domain.

2.2. MD Algorithms

Under normal circumstances, the images that have been processed by the above imaging processing operations still have certain geometric deformations. Therefore, after performing some geometric corrections, the imaging processing can be accomplished to obtain a one-frame image.

In actual ViSAR imaging systems, the motion platform will inevitably generate non-ideal motions, under which circumstances the phase of the radar echo signal will vary, resulting in motion errors. The schematic diagram demonstrating those motion errors is shown in Figure 1. The dashed trajectory represents the ideal motion trajectory of the platform, while the actual trajectory is similar to the motion state described by the solid trajectory.

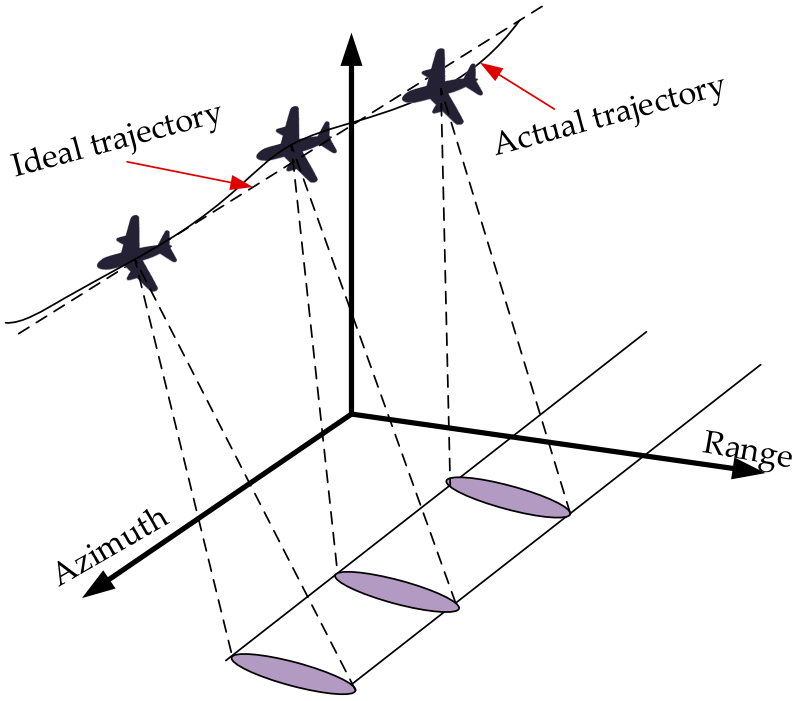


Figure 1. Schematic diagram of motion errors in airborne ViSAR.

In order to solve the problems of Doppler frequency shift and image defocusing caused by motion errors, this study mainly adopts the MD algorithm based on data-driven motion error estimation and compensation to achieve autofocus in real-time ViSAR imaging [28,29]. The core idea of the MD algorithm can be formulated by the following steps. For the MD algorithm, we have made certain improvements to enhance its applicability on FPGA.

The improved MD algorithm first divides the data processed by the RD algorithm into seven sub-blocks of 2048 $\times$ 512 points with an azimuthal overlap rate of 50%. Secondly, the algorithm subdivides these data blocks into 14 sub-sub-blocks of 2048 $\times$ 256 points. Then, conduct zero-padding in the azimuth direction, FFT operation, and azimuth modulus calculation on these sub-sub-blocks.

Subsequently, perform pairwise cross-correlation processing on these 14 sub-sub-blocks and estimate the Doppler frequencies of seven cross-correlated matrices based on the energy distribution of the cross-correlation data. Then, the azimuth frequency of each sub-block is calculated using the method of phase differential differentiation. Finally, concatenate the seven motion errors and perform differential differentiation and least squares fitting to obtain the complete motion error.

The improved MD algorithm is summarized as shown in Algorithm 1.

Algorithm 1: Improved MD Algorithm.

Input:
$RD\ processed\ data: g(x,y), size: N \times N$; $Number\ of\ sub-blocks: N_1$
$Distance\ stereo\ coef: H_1$; $Size\ of\ sub-blocks: N \times N_2$
$Size\ of\ sub-sub-blocks: N \times N_3$
for $i \leftarrow 1$ *to* N_1 **do**
  $t(x,y) \leftarrow g(:,(i-1) \times N_3 + (1:N_2)) \cdot H_1$
  $t_1(x,y) \leftarrow t(:,1:N_3)$
  $t_2(x,y) \leftarrow t(:,N_3 + [1:N_3])$
  $m \leftarrow (N/N_3 - 1)/2$
  for $j \leftarrow 1$ *to* 2 **do**
    $t'_j(x,y) \leftarrow [zeros(N, m \times N_3), t_j(x,y), zeros(N, m \times N_3)]$
    $r_j(x,y) \leftarrow FFT(abs(FFT(t'_i(x,y), dim = 2)), dim = 2)$
  end
  $R(x,y) \leftarrow FFT(abs(IFFT(r_1(x,y) \cdot conj(r_2(x,y)), dim = 2)), dim = 2)$
  $A_1 \leftarrow sum(R(x,y), dim = 1)$
  $A_2[i,:] \leftarrow A_1, t_0 \leftarrow 0, t_1 \leftarrow -1, t_2 \leftarrow 1, n_2 \leftarrow N/2$
  $pos \leftarrow max(A_1, dim = 2)$
  $\phi_{(0,1,2)} \leftarrow A_1(pos + t_{(0,1,2)})$
  $n_1 \leftarrow pos - (\phi_2 - \phi_1)/(2 \cdot (\phi_1 + \phi_2 - 2 \cdot \phi_0))$
  $n \leftarrow (n_1 - n_2)/(2 \times m + 1)$
  $p_1[i,:] \leftarrow -n \times \pi \times ((-N_3 : N_3 - 1)/N_3)^2$
  $p_2[i,:] \leftarrow n \times (N_1 + 1)$
  $s \leftarrow DIFF(-1/(N_1 + 1) \times \pi \times ((-N_3 : N_3 - 1)/N_3)^2)$
end
$phas\_err_1 \leftarrow DIFF(p_1, 1, dim = 2)$
$phas\_err_2 \leftarrow p_2 \cdot (ones(N_1, 1) \cdot s)$
$phas\_err \leftarrow Least\ Squares\ Fitting(phas\_err_1, phas\_err_2)$
$f(x,y) \leftarrow IFFT(IFFT(win(g(x,y) \cdot phas\_err), dim = 1), dim = 1)$;
Output:
$MD\ processed\ data: f(x,y), size: N \times N$

The abovementioned processing flow of the high-frame-rate ViSAR real-time imaging algorithm is illustrated in Figure 2.

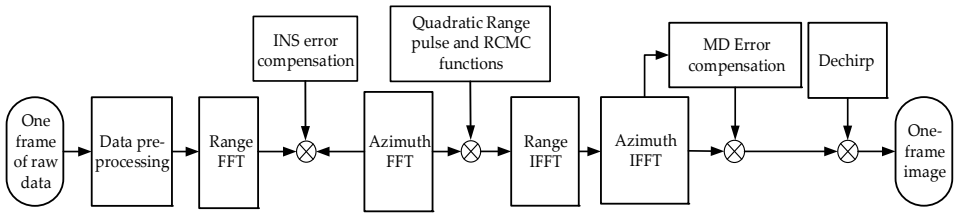


Figure 2. Flow chart of real-time imaging algorithm for ViSAR.

3. Implementing MRIPC on FPGA

3.1. Designing MRIPC Components

The MRIPC component mainly includes a signal generation module, signal processing module, signal acquisition module, data recording module, and control module. The component adopts one high-performance FPGA as the core processor (Model: XC7VX690T

from Xilinx). The high-speed analog-to-digital converter (ADC) and digital-to-analog converter (DAC) chips adopt AD9129 and AD9684, respectively. In order to satisfy the requirement of a 1 GHz bandwidth signal for ViSAR, the transmitting end adopts a 2.4 GHz DAC sampling rate that is generated by double interpolation. The receiving end receives a signal with dechirp modulation, and the bandwidth of the received signal depends on the width of the scene. As for our proposed design in this study, the receiving bandwidth is less than 160 MHz, and the ADC sampling rate is 480 MHz.

With respect to the SAR imaging algorithms, multiple operations require a large amount of data caching, transposition, etc. Unfortunately, the limited block RAM resources within FPGA hinder the fast computation of high-speed data. Particularly when performing parallel computing, efficient access and interaction of the data to be processed have become the bottleneck that restricts the performance of the MRIPC. In response to the above situation, our proposed FPGA architecture in this study, therefore, adopts a design of four sets of 64-bit width DDR3, thereby improving the data access throughput of the MRIPC while providing a larger cache capacity.

Specifically, the physical dimensions of our proposed MRIPC are 145 mm × 95 mm × 6 mm, and its weight is merely 143 g, which is capable of satisfying the requirements of miniaturization and is lightweight for the ViSAR real-time imaging processing components. Figure 3 demonstrates the architecture diagram and the physical image of our proposed MRIPC.

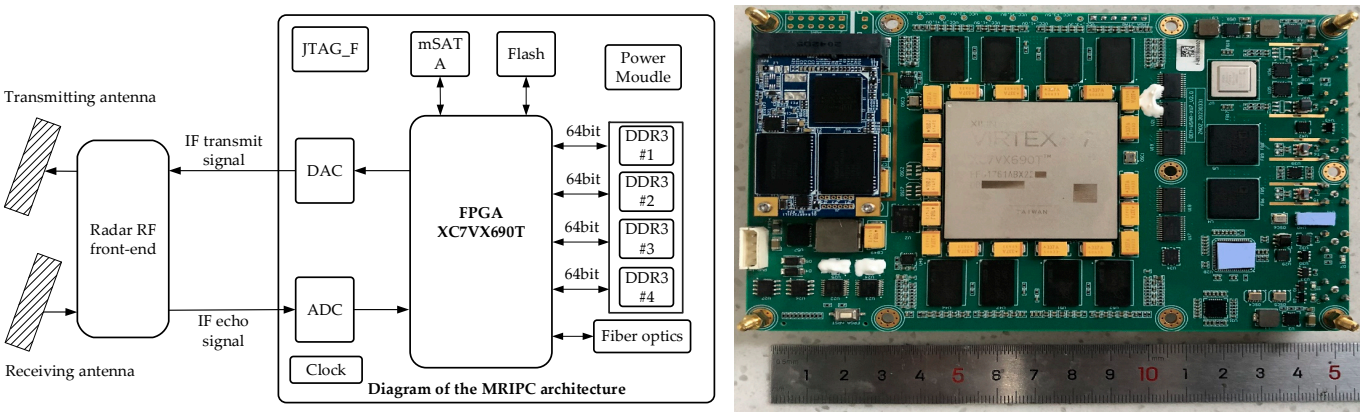


Figure 3. Architecture diagram and physical images of the proposed MRIPC: (**a**) the architecture diagram of the MRIPC; (**b**) the physical image of the MRIPC.

3.2. Implementing High Frame Rate Real-Time Imaging Algorithms on FPGA

3.2.1. Implementing RD Algorithm on FPGA

The implementation process of the RD algorithm on the FPGA primarily involves the range direction and azimuth direction processing, parameter computation and its compensation, as well as data matrix transposition. Figure 4 demonstrates the overall processing flow of the RD algorithm implemented on the FPGA.

When performing FFT and IFFT in the azimuth direction, multiple transposition operations must be conducted with respect to the data matrix. To ensure that the arrangement order of the data flow processed by the RD algorithm remains unchanged, matrix transposition operations must be performed, which will take advantage of the flexibility of the DDR3 storage units, including sequential write and cross-address read, as well as cross-address write and sequential read methods, thereby transposing the matrix.

The FFT operations in the azimuth and range directions underlie the key of the RD algorithm, to which the process of implementing the algorithm on a FPGA is presented as follows.

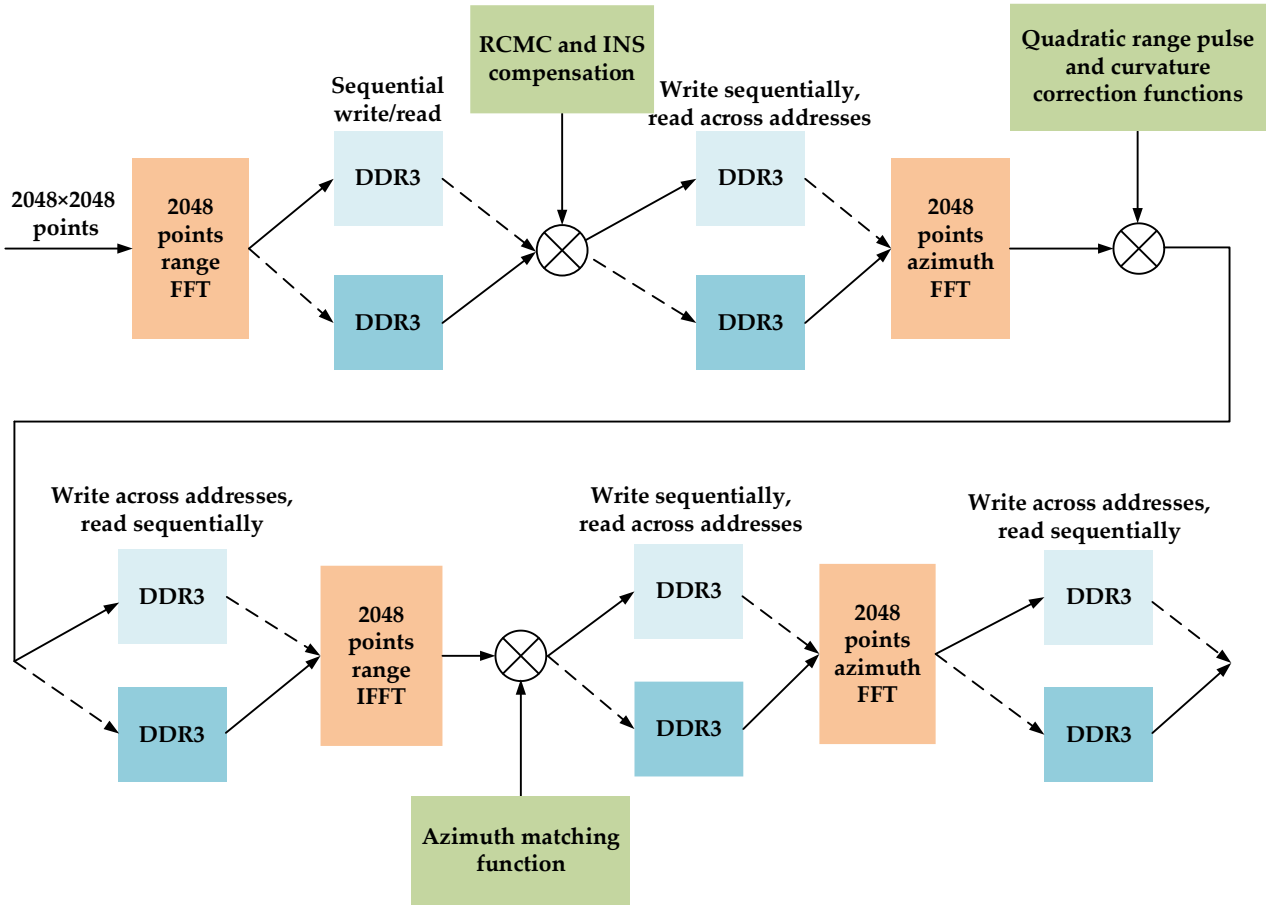


Figure 4. The processing flow of implementing RD algorithm on FPGA.

Step 1. Perform FFT in the range direction. When the data of one frame of 2048 × 2048 single-precision floating-points are preprocessed, perform FFT with respect to it in the range direction, during which all of the FFT operations adopt a pipeline parallel computation mode to improve the computation efficiency. Then, the processed data matrix is sequentially written into the DDR3 storage. In this study, all the DDR3 cache units used adopt a ping-pong design, which thus trades off the storage resources for processing speed to satisfy the requirements of pipeline processing.

Step 2. Correct the range cell walk values and compensate for INS errors. Use a DSP48 multiplier to perform a matrix dot product on the data matrix to which the FFT operations have been conducted in the range direction with the INS compensation parameters and with range cell walk correction parameters to obtain the compensated data, and then write it sequentially into the DDR3 storage unit.

Step 3. Perform FFT in the azimuth direction. Use the DDR3 storage unit for cross-address read and write. When the data matrix is transposed, conduct FFT in the azimuth direction with respect to the processed data matrix.

Step 4. Perform IFFT in the range direction. The data matrix that has been compensated by the functions of quadratic range pulse compression and range curvature correction is written into the DDR3 storage unit across addresses in the azimuth direction. Then, sequentially read the written data in the range direction, transpose the data matrix again, and conduct IFFT in the range direction on the matrix. The data to which the IFFT is conducted in the range direction are then compensated with the azimuth matching function by means of pipeline processing.

Step 5. Perform IFFT in the azimuth direction. After being compensated by the azimuth matching function, the data matrix is first sequentially written into the DDR3 storage

unit and then read out across addresses to realize transposition, thereby transforming them from the range direction to the azimuth direction.

All the FFT operations are provided by exclusively designed FFT_IP cores that are configured in a pipeline processing mode to divide the computational tasks into a series of independent operations, thereby achieving synchronous execution of multiple tasks while effectively improving the overall computing speed. Considering the limitations of the hardware resources and memory, the processing unit that performs FFT and IFFT processing in the azimuth direction is organized into a parallelized operation group consisting of eight independent IP cores, which can provide eight-way parallel FFT and IFFT operations. It completes the compensation processing of the quadratic range pulse compression function and range correction function through the same parallel computation.

3.2.2. Implementing MD Algorithm on FPGA

Despite that, the frequency-domain RD imaging algorithm possesses the advantage of low computation complexity; however, defocusing occurs to the image obtained after being processed by the RD algorithm in the frequency domain, for which reason motion error estimation and compensation are required.

When the data processed by the RD algorithm enters into the motion error estimation and compensation unit, the data will be divided into two separate ways. One way the data is used is for motion error estimation, while the other is used as raw data for motion error compensation. The sub-block segmentation module subdivides the data matrix into seven sub-blocks in the azimuth direction to facilitate the estimation of Doppler frequency modulation. The sub-block processing module contains seven independent processing components for processing the segmented sub-blocks, which can generate seven sets of Doppler frequency modulation. The motion error estimation module estimates and concatenates seven sets of motion error data and fits them into one set of motion error data. As for the motion compensation module, it is primarily used to compensate for the estimated motion error in the data processed by the RD algorithm, thereby eliminating the Doppler motion error in the data.

During the implementation process of the MD algorithm on the FPGA, we adopted a parallel design to deal with the sub-blocks. If a seven-way fully parallel processing scheme is adopted, 14 sub-block processing operations are to be executed on the FPGA, involving 28 FFT processing cores in the azimuth direction. In addition, after performing azimuth cross-correlation processing in the azimuth direction on the sub-blocks in the frequency domain, it is necessary to perform IFFT operations in the azimuth direction on the cross-correlated data blocks and then to transform the resultant data back into the time domain. During this process, the extra seven IFFT processing cores in the azimuth direction are required. It must be noted that adopting the seven-way parallel processing scheme will result in a high occupancy rate of FPGA resources, which not only increases the synthesis difficulty for the FPGA but also lowers the wiring efficiency and makes it difficult to satisfy the internal timing constraints of the FPGA.

In order to strike a balance between computing speed and hardware resources, the proposed scheme in this study improved the working frequency of the sub-block processing unit of the MD algorithm to twice that of the RD processing unit, divided the data into seven sub-blocks in the azimuth direction, and replaced the seven-way fully parallel processing scheme by reusing four sub-block parallel processing units. Meanwhile, for the purpose of saving hardware resources, the proposed scheme reused the processing modules.

Since the overall working frequency of the RD processing unit is 120 MHz, the sub-block parallel processing unit designed in this study shall have a working frequency of 240 MHz in the MD algorithm. When sub-block processing is accomplished, data concatenation is performed, and the subsequent processing speed should regress to 120 MHZ Since the increased working frequency of the sub-block processing unit is 240 MHz, which is twice that of the RD processing unit, the real-time design requirements are thus satisfied, like the seven-way fully parallel processing scheme.

When implementing the MD algorithm on the FPGA, the first step is to utilize FPGA logic to segment the 2048 × 2048 points data processed by the RD algorithm into seven sub-blocks with 2048 × 512 points, with an overlap rate of 50% between each adjacent sub-block. Then, four groups of sub-blocks with 2048 × 256 × 5 points are input into the sub-block processing unit. Meanwhile, the remaining three sets of sub-blocks with 2048 × 256 × 3 points are stored in the DDR3. Finally, after processing the first four groups of sub-block data, they are cached in the concatenation processing unit for processing. Before these data completely enter into the pipeline parallel processing unit, the last three groups of sub-block data are then sent to the parallel processing unit for processing.

In the motion error estimation and compensation unit, the data block is first segmented from 2048 × 2048 points into seven sub-blocks of 2048 × 512 points with an overlap rate of 50% between each adjacent block. Then, each of the above sub-blocks is further subdivided into two sub-sub-blocks of 2048 × 256 points, to which zero-padding is conducted in the azimuth direction to make the size of each sub-sub-block size 2048 × 2048 points. The above process can be directly implemented through FPGA logic without requiring DDR3 cache resources.

The processing steps for these 14 sub-sub-blocks are formulated as follows.

Step 1. Perform FFT, respectively, on the 14 sub-sub-blocks in the azimuth direction to obtain the result of 2048 points in the azimuth direction, then take the modulus value of the result.

Step 2. Perform frequency-domain cross-correlation processing on each of the final results of the above step to obtain seven cross-correlated data blocks with 2048 × 2048 points.

Step 3. Perform IFFT on each cross-correlated data block in the azimuth direction, transforming the data into a time domain to obtain seven time-domain cross-correlated data blocks.

Step 4. Perform summation on each of the above time-domain cross-correlated data blocks in the range direction to obtain seven real cross-correlated data blocks with 1 × 2048 points.

Step 5. Find the maximum value of each of the above data blocks in the azimuth direction, expand the data block to seven data blocks with 1 × 512 points, and concatenate these seven data blocks.

Step 6. Perform differential differentiation and least squares fitting on the concatenated data block to obtain the motion error-compensated data with 2048 × 2048 points.

Step 7. Perform windowing and IFFT on the above-compensated data in the range direction and then compensate it to the data processed by the RD algorithm to complete the imaging processing.

Since the entire process of data processing is conducted in the azimuth direction, only one DDR3 storage unit is required for sub-block multiplexing and parallel processing, whereas other processing units are not required to be cached in the DDR3 storage.

The specific processing flow of implementing the MD algorithm on the FPGA is demonstrated in Figure 5.

In order to verify the advantage of our proposed four-way parallel multiplexing processing scheme, we performed the RD and MD algorithms using a seven-way fully parallel scheme and four-way parallel multiplexing scheme, respectively, on a single-chip FPGA and recorded the resource utilization rates under the two processing schemes. Figure 6 demonstrates the comparison results of the utilization rates of the main resources under the two schemes.

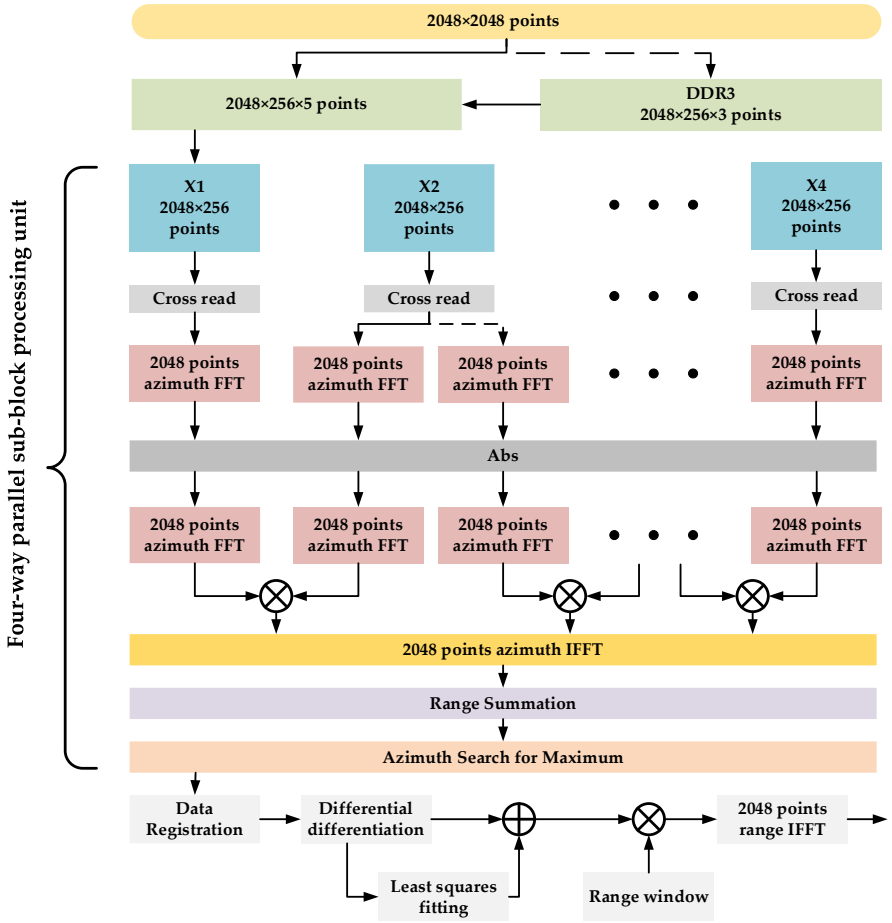


Figure 5. The processing flow of implementing the MD algorithm on FPGA.

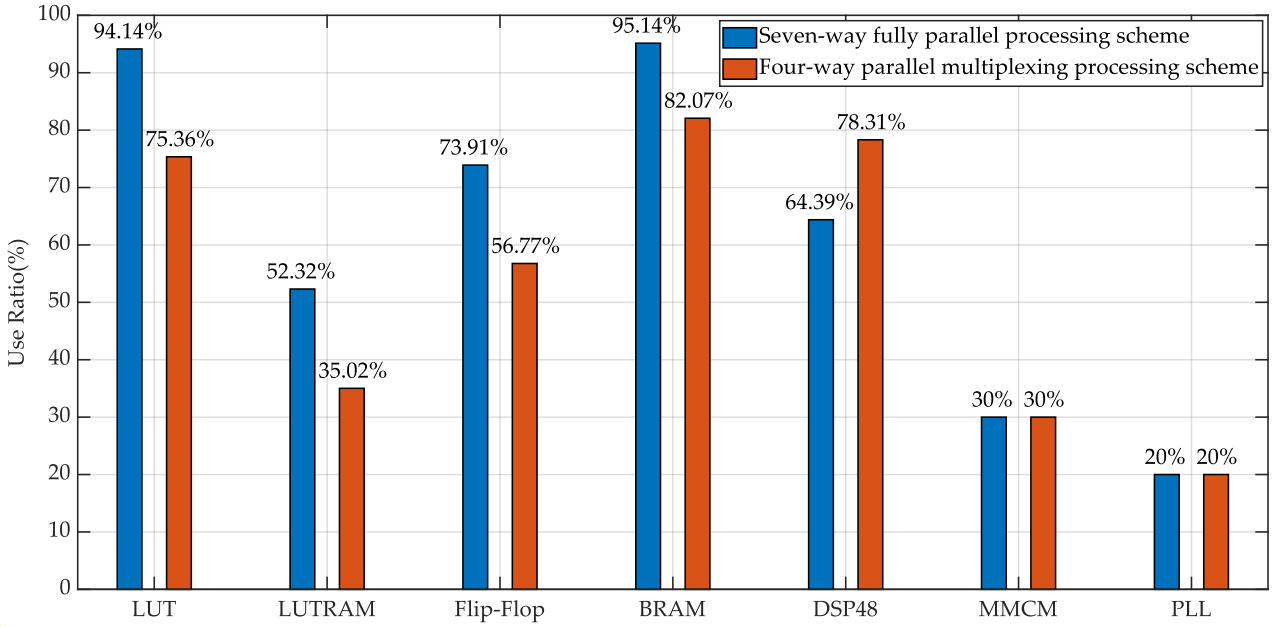


Figure 6. Comparison results of resource utilization rates of the two processing schemes.

It can be observed from Figure 6 that under the seven-way fully parallel scheme, both the LUT resources and BRAM occupancy rates exceed 90%, which can hardly be realized during the layout and wiring, not to mention the tight timing. Meanwhile, the LUTRAM

and Flip-Flop resources also confronted great pressure. In contrast, our proposed four-way parallel multiplexing processing scheme regulates the resource occupancy within a relatively reasonable range. This architecture optimizes the balance between the hardware resources and processing speed, allowing a single FPGA chip to efficiently execute real-time ViSAR imaging algorithms.

4. Verification of MRIPC and Analysis

4.1. Processing Result of MRIPC

In order to verify the processing result of the proposed MRIPC in this study, we conducted real-time ViSAR imaging processing using measured data. The airborne radar operated in the strip-map mode, and we replayed multiple frames of radar data collected at an echo rate through a data recorder, after which these data were processed on the MRIPC platform. The design of the above procedure aims to simulate airborne real-time processing conditions, through which we can conduct an effective performance evaluation under conditions similar to actual application scenarios. Figure 7 shows the real-time imaging results of nine consecutive frames.

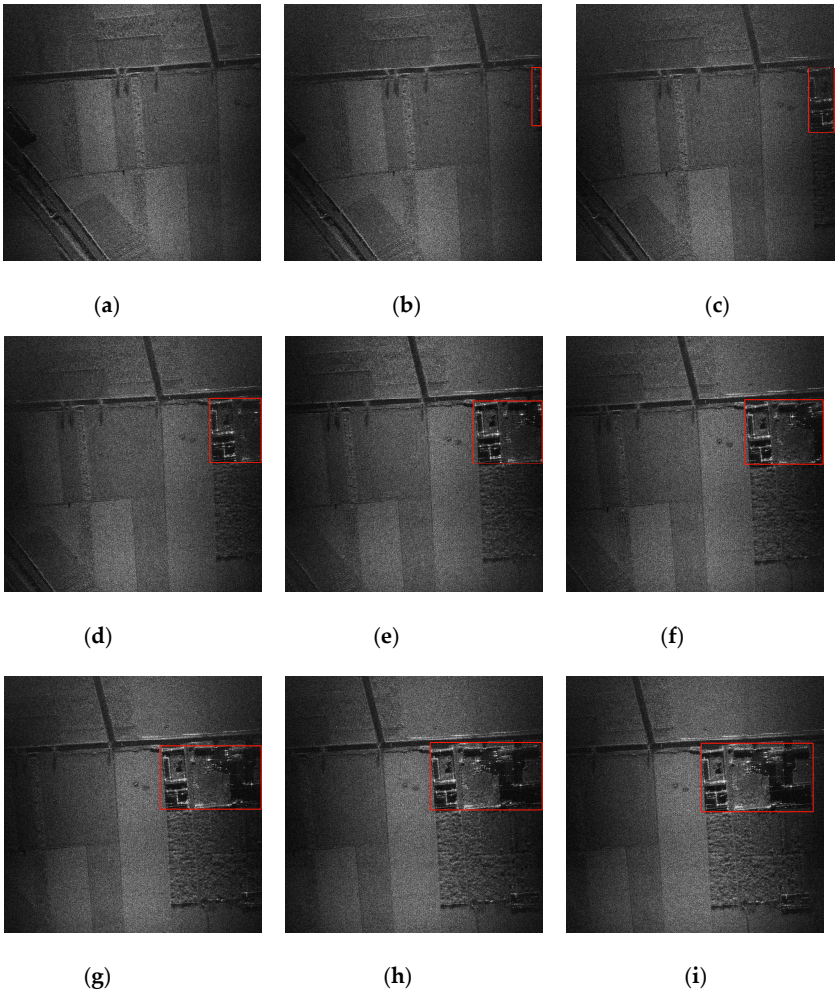


Figure 7. Real-time imaging results of ViSAR based on the proposed MRIPC: (**a**) The first frame image; (**b**) the second frame image; (**c**) the third frame image; (**d**) the fourth frame image; (**e**) the fifth frame image; (**f**) the sixth frame image; (**g**) the seventh frame image; (**h**) the eighth frame image; and (**i**) the ninth frame image.

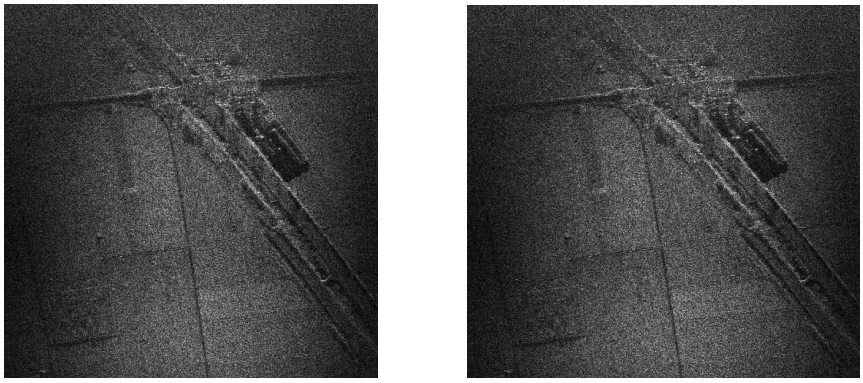
Figure 7 shows the proposed MRIPC real-time imaging results of nine consecutive frames, processed with respect to the provided 2048 $\times$ 2048 single-precision floating-point

data, where the slowly emerging farmland is highlighted in red boxes. The total time for processing nine consecutive frames of data is 1.8 s, and the imaging time for each frame of data is 0.2 s. The test results satisfy the requirement of a real-time imaging frame rate of 5 Hz. In actual testing, the proposed MRIPC scheme can satisfy the requirement of continuous imaging without losing scenes.

On the proposed MRIPC platform, through BackPressure, we regulated that the processing of one frame of 2048 × 2048 single-precision floating-point data can be completed within 200 ms. This mechanism ensures that when the rate of input data is too high, the flow control logic can pause the input data until the RD and MD algorithms have completed processing and are ready to accept new data. The adoption of this approach avoids data accumulation and guarantees that for each frame of data, the image processing can be completed within 200 ms by adaptively adjusting the data input rate.

4.2. Processing Performance of MRIPC

In order to verify the processing performance of our proposed MRIPC, we used a data recorder to replay the collected measured data for processing. The original data are 2048 × 2048 points of single-precision floating-point data, which were processed on the MRIPC platform using single-chip FPGA reconstruction-accelerated RD and MD algorithms. Meanwhile, we used MATLAB 2023a software to run the traditional RD and MD algorithms to process the same data. The two different processing methods were used to obtain one frame of ViSAR imaging results, as shown in Figure 8.



(**a**) (**b**)

Figure 8. One-frame imaging results of ViSAR: (**a**) MRIPC and (**b**) MATLAB.

To study the visual quality variations in the ViSAR imaging frames during the real-time processing of the proposed MRIPC, we used two indicators, i.e., the peak signal-to-noise ratio (PSNR) and the structure similarity index measure (SSIM), to compare the image quality before and after accelerated processing. We also compared the proposed processing scheme with research related to ViSAR, to which the results are listed in Table 1. It can be seen that, compared with the previous related studies, our proposed processing scheme exhibits a shorter processing time and higher imaging quality.

Table 1. Comparison results of processing schemes.

Work	Platform	Image Size	Algorithm	Time (s)	Operating Frequency	Imaging Quality
[30]	Xilinx XC7VX690T	900 × 900	BP	1.10 s	200 MHz	PSNR = 27.26 dB SSIM = 0.8652
[31]	Xilinx Zynq UltraScale + FPFA	2048 × 2048	CS	0.48 s	235 MHz	PSNR = 33.43 dB SSIM = 0.947

Table 1. *Cont.*

Work	Platform	Image Size	Algorithm	Time (s)	Operating Frequency	Imaging Quality
[32]	Jetson AGX Orin	2048 × 2048	RD + MD	0.135 s	1.30 GHz	PSNR = 48.1308 dB SSIM = 0.9652
Ours	Xilinx XC7VX690T	2048 × 2048	RD + MD	0.20 s	200 MHz	PSNR = 37.64 dB SSIM = 0.954

5. Conclusions

The UAV-borne ViSAR imaging systems exhibit characteristics of high data throughput and high frame-rate imaging, imposing considerable demands on the real-time processing capabilities of signal processing systems. Furthermore, to satisfy the requirements of UAV integration, higher integration, miniaturization, and lower power consumption are also necessary. Currently, onboard research regarding UAV-borne ViSAR systems mainly focuses on the DSP, embedded GPU, and FPGA platforms. The DSP platform adopts a serial processing approach, which is not suitable for high-frame-rate real-time imaging in ViSAR systems. In comparison to the FPGA, although embedded GPUs can realize higher imaging efficiency and higher image quality, they are limited by the GPU interfaces and cannot directly communicate with ADCs and DACs. They require FPGAs or similar chips for radar waveform generation and signal acquisition interface conversion. Furthermore, data transmission through the PCIe limits the real-time processing capability of ViSAR imaging on embedded GPU platforms. In addition, the flexibility and scalability of embedded GPUs are low, making it difficult to effectively increase integration while reducing the system's size and weight.

In response to the above problems, this study proposed a miniaturized design of a highly integrated real-time imaging processing component for the UAV-borne ViSAR. Firstly, aiming at realizing the goals of miniaturization, high data throughput, and real-time imaging processing, the proposed component integrates the main functions of a ViSAR system within a single large-scale FPGA chip, thereby realizing a miniaturized ViSAR module with 145 mm × 95 mm × 6 mm physical dimensions and a 143 g weight. Secondly, our proposed component adopts a parallel access mechanism with multiple sets of high-speed data buffers to improve the data access throughput. Thirdly, we adopted a parallel multiplexing strategy to optimize the RD and MD algorithms, by which a single FPGA chip can efficiently execute real-time ViSAR imaging algorithms. The test results have verified the effectiveness of our proposed component in real-time processing of 2048 × 2048 points single-precision floating-point data, achieving a 5 Hz imaging frame rate and 0.15 m imaging resolution. Compared to previous studies on GPUs and the FPGA, our proposed component takes the shortest time to generate a ViSAR image with a higher PSNR and SSIM. In addition, combining the advantages of miniaturization and high data throughput, our proposed component provides an effective alternative solution for UAV-borne ViSAR real-time imaging processing components.

Despite that, adopting the FPGA platform can significantly improve system integration; however, FPGA programming and debugging still remain great challenges, let alone its long development cycle. However, in order to achieve a high integration of ViSAR systems on a single chip, the FPGA stands out as a favorable option. In addition, due to limited FPGA resources, the processing efficiency of complex ViSAR imaging algorithms performed on FPGA can hardly be improved.

ViSAR technology is developing towards miniaturization and software, and there are currently heterogeneous chips similar to the RFSoC that can improve the overall performance and real-time performance of imaging processing components. Our future research interest will be focusing on this integrated architecture to design real-time imaging processing components for the UAV-borne ViSAR, which can improve processing performance while reducing costs. Meanwhile, our proposed component merely designs motion error estimation and compensation methods based on the RD imaging algorithm and MD algo-

rithm. Our future work will focus on the implementation of various imaging algorithms for ViSAR, including but not limited to the BP algorithm, PFA algorithm, and PGA algorithm. In the case of PGA algorithms with high computational complexity, we will be planning to employ a hybrid approach utilizing an embedded GPU in conjunction with the FPGA. This approach involves porting computationally intensive operations to embedded GPUs to fully leverage their respective strengths, enhancing the real-time imaging efficiency for ViSAR. Concurrently, potential application scenarios for the ViSAR system will be thoroughly explored, with the aim of successful implementation in various complex environments, in which the scalability of the system will be rigorously tested and evaluated to fully demonstrate the practical value and applicability of the proposed system.

Author Contributions: Conceptualization, T.Y.; methodology, T.Y., T.W. and N.Z.; software, N.Z. and F.M.; validation, T.W., F.M. and X.Z.; formal analysis, T.Y. and S.Z.; investigation, T.W. and F.M.; resources, T.Y.; data curation, N.Z. and X.Z.; writing—original draft preparation, T.Y., T.W. and Q.W.; writing—review and editing, T.Y., Q.W. and X.Z.; visualization, T.W.; supervision, T.Y. and S.Z.; project administration, T.Y.; funding acquisition, T.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by the Key Research and Development Program of Shaanxi, under grant 2024GX-YBXM-160.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Cao, N.; Lee, H.; Zau, E. Estimation of Residual Motion Errors in Airborne SAR Interferometry Based on Time-Domain Back projection and Multisquint Techniques. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 2397–2407. [CrossRef]
2. An, H.; Wu, J.; Teh, K.; Sun, Z.; Li, Z.; Yang, J. Joint Low-Rank and Sparse Tensors Recovery for Video Synthetic Aperture Radar Imaging. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5214913. [CrossRef]
3. Yan, H.; Liu, H.; Zhou, Y.; Cheng, L. A New Method of Video SAR Ground Moving Target Detection and Tracking Based on the Interframe Amplitude Temporal Curve. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 5219217. [CrossRef]
4. Li, Z.; Qiu, X.; Yang, J.; Meng, D.; Huang, L.; Song, S. An Efficient BP Algorithm Based on TSU-ICSI Combined with GPU Parallel Computing. *Remote Sens.* **2023**, *15*, 5529. [CrossRef]
5. A Edwards, M.; Madsen, D.; Stringham, C.; Margulis, A.; Wicks, B.; Long, D.G. microASAR: A small, Robust LFM-CW SAR for Operation on UAVs and Small Aircraft. In Proceedings of the 2008 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Boston, MA, USA, 7–11 July 2008; pp. 514–517. [CrossRef]
6. MiniSAR. Available online: http://www.sandia.gov/radar/images/SAND.2014.8 (accessed on 4 November 2023).
7. Halcrow, G.; Greig, D.W.; Glass, A. PicoSAR trials results. In Proceedings of the 2013 14th International Radar Symposium (IRS), Dresden, Germany, 19–21 June 2013; pp. 47–52.
8. Wiehle, S.; Mandapati, S.; Günzel, D.; Breit, H.; Balss, U. Synthetic Aperture Radar Image Formation and Processing on an MPSoC. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5226814. [CrossRef]
9. Yang, L.; Zhao, L.; Zhou, S.; Bi, G.; Yang, H. Spectrum-Oriented FFBP Algorithm in Quasi-Polar Grid for SAR Imaging on Maneuvering Platform. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 724–728. [CrossRef]
10. Li, Y.; Xu, G.; Zhou, S.; Xing, M.; Song, X. A Novel CFFBP Algorithm with Noninterpolation Image Merging for Bistatic Forward-Looking SAR Focusing. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5225916. [CrossRef]
11. Song, Y.; Hai, Y.; Wu, J.; Li, Z.; Yang, J. An Efficient PFA Sub aperture Algorithm for Video SAR Imaging. In Proceedings of the 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, Brussels, Belgium, 11–16 July 2021; pp. 5179–5182. [CrossRef]
12. Zhang, Q.; Wu, J.; Li, Z.; Miao, Y.; Huang, Y.; Yang, J. PFA for Bistatic Forward-Looking SAR Mounted on High-Speed Maneuvering Platforms. *IEEE Trans. Geosci. Remote Sens.* **2019**, *57*, 6018–6036. [CrossRef]
13. Wang, Z.; Wei, F.; Huang, Y.; Zhang, X.; Zhang, Z. Improved RD imaging method based on the principle of step-by-step calculation. In Proceedings of the 2021 2nd China International SAR Symposium (CISS), Shanghai, China, 3–5 November 2021; pp. 1–5. [CrossRef]
14. Gui, L.; Hai, Y.; Wu, J.; Li, Z.; Yang, J. A Motion Error Estimation Method of UWB-SAR Based on Coherent Correlation Function. In Proceedings of the 2022 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Kuala Lumpur, Malaysia, 17–22 July 2022; pp. 2083–2086. [CrossRef]
15. Chen, J.; An, D.; Wang, W.; Chen, L.; Feng, D.; Zhou, Z. A Novel Generation Method of High Quality Video Image for High Resolution Airborne ViSAR. *Remote Sens.* **2021**, *13*, 3706. [CrossRef]

16. Chen, J.; Li, M.; Yu, H.; Xing, M. Full-Aperture Processing of Airborne Microwave Photonic SAR Raw Data. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 1–12. [CrossRef]
17. Wang, T.; Zhu, D.; Meng, X. Implementation of Terahertz Video SAR Imaging Based on Multi-Core DSP. In Proceedings of the 2022 7th International Conference on Signal and Image Processing (ICSIP), Suzhou, China, 20–22 July 2022; pp. 541–545. [CrossRef]
18. Zhang, Y.; Zhou, J.; Song, Z.; Zhou, K. High-Precision GPU-Accelerated Simulation Algorithm for Targets under Non-Uniform Cluttered Backgrounds. *Remote Sens.* **2023**, *15*, 4664. [CrossRef]
19. Tian, H.; Hua, W.; Gao, Y.; Sun, Z.; Cai, M.; Guo, Y. Research on Real-time Imaging Method of Airborne SAR Based on Embedded GPU. In Proceedings of the 2022 3rd China International SAR Symposium (CISS), Shanghai, China, 2–4 November 2022; pp. 1–4. [CrossRef]
20. Yang, T.; Xu, Q.; Meng, F.; Zhang, S. Distributed Real-Time Image Processing of Formation Flying SAR Based on Embedded GPUs. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2022**, *15*, 6495–6505. [CrossRef]
21. Cai, M.; Wang, H.; Hua, W. Research on Optimal Design of Spaceborne SAR Real-time Imaging Technology Based on FPGA. In Proceedings of the 2021 2nd China International SAR Symposium (CISS), Shanghai, China, 3–5 November 2021; pp. 1–2. [CrossRef]
22. Hettiarachchi, D.L.N.; Balster, J.E. Fixed-Point Processing of the SAR Back-Projection Algorithm on FPGA. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 10889–10902. [CrossRef]
23. Tong, W.; Wei, B.; Yu, A.; Dong, Z.; He, Z.; Tang, F. Video SAR Moving Target Detection System Based on FPGA. In Proceedings of the 2023 8th International Conference on Signal and Image Processing (ICSIP), Wuxi, China, 8–10 July 2023; pp. 419–423. [CrossRef]
24. Choi, Y.; Jeong, D.; Lee, M.; Lee, W.; Jung, Y. FPGA Implementation of the Range-Doppler Algorithm for Real-Time Synthetic Aperture Radar Imaging. *Electronics* **2021**, *10*, 2133. [CrossRef]
25. Mota, D.; Cruz, H.; Miranda, P.R.; Duarte, R.P. Onboard Processing of Synthetic Aperture Radar Back-projection Algorithm in FPGA. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2022**, *15*, 3600–3611. [CrossRef]
26. Chen, J.; Liang, B.; Zhang, J.; Yang, G.D. Efficiency and Robustness Improvement of Airborne SAR Motion Compensation with High Resolution and Wide Swath. *IEEE Geosci. Remote Sens. Lett.* **2022**, *19*, 4004005. [CrossRef]
27. Li, C.; Zhang, H.; Deng, Y.; Wang, R. Focusing the L-Band Spaceborne Bistatic SAR MissionData Using a Modified RD Algorithm. *IEEE Trans. Geosci. Remote Sens.* **2020**, *58*, 294–306. [CrossRef]
28. Jin, Y.; Chen, J.; Xia, X.G. Ultrahigh-Resolution Autofocusing for Squint Airborne SAR Basedon Cascaded MD-PGA. *IEEE Geosci. Remote Sens. Lett.* **2022**, *19*, 4017305. [CrossRef]
29. Tang, Y.; Zhang, B.; Xing, M.; Bao, Z.; Guo, L. The Space-Variant Phase-Error Matching Map-Drift Algorithm for Highly Squinted SAR. *IEEE Geosci. Remote Sens. Lett.* **2013**, *10*, 845–849. [CrossRef]
30. Cao, Y.; Guo, S.; Jiang, S.; Zhou, X.; Wang, X.; Luo, Y.; Yu, Z.; Zhang, Z.; Deng, Y. Parallel Optimisation and Implementation of a Real-Time Back Projection (BP) Algorithm for SAR Based on FPGA. *Sensors* **2022**, *22*, 2292. [CrossRef]
31. Lee, J.; Jeong, D.; Lee, S.; Lee, M.; Lee, W.; Jung, Y. FPGA Implementation of the Chirp-Scaling Algorithm for Real-Time Synthetic Aperture Radar Imaging. *Sensors* **2023**, *23*, 959. [CrossRef] [PubMed]
32. Yang, T.; Zhang, X.; Xu, Q.; Zhang, S.; Wang, T. An Embedded GPU-Based Scheme for Real-Time Imaging Processing of Unmanned Aerial Vehicle Borne Video Synthetic Aperture Radar. *Remote Sens.* **2024**, *16*, 191. [CrossRef]