An FPGA-based real-time UAV SAR raw signal simulator

Wei Li^{1a)}, Houxiang Zhang¹, Hans Petter Hildre¹, and Jun Wang²

¹ Alesund University College

² Beihang University

a) windriver@126.com

Abstract: Faced with the high computational complexity of UAV SAR raw signal simulators, a multi-FPGA system is developed. The system is based on a time-domain raw signal algorithm which can compute in real time and can be used for closed-loop simulation. In order to improve the efficiency of the SAR slant range computing, a modified non-restoring square root algorithm for FPGA is designed. An improved method is presented to perform coherent accumulation of raw signal to decrease memory cost. The pipelined FFT and IFFT are used to compute the convolution in order to reduce delay. The SAR raw signal generation system is implemented and verified with real-time performance. It can simulate an imaging scene size of 640 * 640 point scatters with a PRF higher than 40 kHz.

Keywords: FPGA, SAR, UAV

Classification: Integrated circuits

References

- [1] I. G. Cumming and F. H. Wong: *Digital Processing of Synthetic Aperture Radar Data* (Artech House, Norwood, MA, 2005).
- [2] A. Mori and F. De Vita: *IEEE Trans. Geosci. Rem. Sens.* **42** (2004) 1811. DOI: [10.1109/TGRS.2004.832242](https://doi.org/10.1109/TGRS.2004.832242)
- [3] Z. Xujin and Z. Zhaoda: *Modern Radar* **29** (2007) 9.
- [4] A. DeHon: *IEEE Comput.* **33** (2000) 41. DOI: [10.1109/2.839320](https://doi.org/10.1109/2.839320)
- [5] L. Yamin and C. Wanming: *IEEE Symposium on FPGA for Custom Computing Machines* (1997) 226. DOI: [10.1109/FPGA.1997.624623](https://doi.org/10.1109/FPGA.1997.624623)
- [6] T. Sutikno: *Int. J. Comput. Theor. Eng.* **3** (2011) 46. DOI: [10.7763/IJCTE.2011.V3.281](https://doi.org/10.7763/IJCTE.2011.V3.281)
- [7] S. Samavi, A. Sadrabadi and A. Fanian: *J. Systems Archit.* **54** (2008) 957. DOI: [10.1016/j.sysarc.2008.04.004](https://doi.org/10.1016/j.sysarc.2008.04.004)
- [8] S. He and M. Torkelson: *ISSSE* (2001) 257. DOI: [10.1109/ISSSE.1998.738077](https://doi.org/10.1109/ISSSE.1998.738077)

1 Introduction

Synthetic Aperture Radar (SAR) systems are special types of radar that produce high resolution images (comparable to optical sensors) in all weather conditions, night and day. SAR sensors can either be mounted on space-crafts

(space-borne systems) or placed on airplanes (airborne systems) [1]. SAR systems utilise sophisticated signal processing and image processing algorithms and use complicated radar electronics and processing units. Requirements, specifications, design and testing of such systems are complicated and costly.

In recent years, a new remote sensing technology based on Unmanned Aerial Vehicle (UAV) SAR has emerged. This technology provides a great potential for detailed monitoring and surveillance of areas covering of up to a few thousand square kilometres. When compared to manned aircraft, the advantages of using a UAV are that it does not need a qualified pilot on board, can enter environments that are dangerous for humans, can be programmed to work autonomously, performs a precise scan of a region during day and night and it can be built at a relatively low cost. One of the biggest challenges of this UAV SAR system is the loss of image quality due to motion errors. Because of the size and the weight of the UAV, it is very difficult to steadily manoeuvre in cloudy and windy conditions. The flight path may have significant trajectory deviations from the ideal path and variations in the UAV's forward velocity and attitude (yaw, pitch, and roll) further complicate the problem. In order to test and verify the functionality and performance of the UAV SAR, simulation is in great need.

The SAR raw signal simulation tool is a reliable solution to reduce such costs and complexity. Several works on SAR raw signal simulation have been published in recent years [2]. Most of them concern using the algorithm to generate the raw signal. In order to verify and test the radar capabilities, a hardware-in-the-loop (HWIL) simulator is very useful. Most SAR HWIL simulators are non real-time [3]. The raw signal data is generated based on a predefined trajectory and loaded in a hardware system to be converted to analog signal. The main shortcoming of this type of simulator is that it is difficult to perform closed-loop system testing and fully verify the performance of the system. The flight trajectory of a highly dynamic platform such as a UAV is hard to define but can be tested in a closed-loop system with a turntable on which the platform can take a specific trajectory.

Standard SAR processing techniques are based on the assumption that the sensor flight path is a straight line. This is usually a good approximation for space-borne sensors. In this case, using the SAR impulse response in a frequency domain (FD) leads to a computationally efficient simulator. This allows us to obtain the raw signal from extended targets with a low CPU time. A time-domain (TD) raw simulator, evaluating a coherent accumulation of the target echo for each transmitted radar impulse, can easily consider the real trajectory of the platform and other effects such as mechanical structure oscillation. The main drawback of a TD simulator is the very high computational complexity and is often used with geometrically limited targets.

In order to address the high computational complexity of UAV SAR raw signal simulators, a multiprocessor system is developed. At present, there are many computing devices including FPGA (Field-programmable gate-array), DSP (Digital Signal Processing), ASIC (Application Specific Integrated

Circuit) and distribution network parallel computing [4]. FPGA and DSP devices are preferred for real time computing. When compared with FPGA, DSP lacks the high degree of parallelism with limited computing and I/O resources and therefore is not suitable for parallel computing applications. FPGA devices are widely used in signal processing, communications, and network applications because they are reconfigurable and support parallelism. The disadvantage is that the development cycle of the FPGA design is usually longer than the DSP implementation. But once an efficient architecture is developed and the parallel implementation is explored, FPGA is able to significantly improve the processing speed because of its intrinsic density advantage.

In this paper, an FPGA based SAR raw signal simulator is implemented. It is different from most SAR signal generators using predefined radar platform parameters and stored raw data. This simulator can receive ever changing radar parameters and generate the raw signal according to the radar parameters in real time. Multiple FPGAs are used in the simulator where the FPGA is the main device responsible for raw signal computing with fixed-point format. In order to achieve real-time processing, some optimisation is presented. First, a slant range computing algorithm is presented which is optimised for an FPGA with fixed point format. Second, the memory architecture for return data coherent accumulation is optimised for an FPGA. Third, the pipelined FFT processor is presented for fast convolution.

2 Algorithm description

In this section, we recall the rationale of the SAR raw signal simulation, as implemented in [2]. Let us consider a radar system located on a platform illuminating the earth's surface. In each azimuth position (spaced of $1/\text{PRF}$ seconds), the echoes from each surface scatters are coherently added, considering their proper phase factor and time delay. After considering a null video-offset frequency, the simulated SAR signal received of every PRF (pulse repetition frequency) is given by Eq. (1).

$$s_r = \sum_{i,j} G_{ij} RCS_{ij} \exp[-j4\pi R_s/\lambda] \cdot \delta[r - R_s] \otimes \frac{2}{cv} s_t \quad (1)$$

$i = 0, 1, \dots, N_a - 1$, $j = 0, 1, \dots, N_r - 1$, N_a, N_r , the number of scatters in azimuth and range directions, respectively. G_{ij} , the pattern antenna weight, λ , the wavelength, RCS_{ij} , the complex backscattering coefficient of scatter of point (i,j), R_s , the slant range between a point scatter and the antenna phase centre, $\delta(\cdot)$, the pulse envelope, c , the velocity of light, v , the velocity of the platform and s_t , the transmit signal. The raw signal data is the result of the convolution between the accumulation of the return signals of the scatters and the transmit signal.

The raw data generation procedure is based on the block diagram shown in Fig. 1. The steps are: 1) Receiving the simulation or test parameters, such as the platform position, velocity; 2) Computing the slant range of scatters in the scene; 3) computing the azimuth phase and multiplying it by the Radar

Cross Section (RCS); 4) accumulation of the return signal of the same range cell; 5) convolution with the transmit signal; 6) conversion of the digital signal to analog signal.

The radar motion parameters can be received from radar test platform in a HWIL system or can be determined by the controller in this simulator. The radar motion parameters include radar platform position, velocity, antenna attitude. The parameters update rate ranges from 100 Hz to 1 kHz and is usually slower than PRF. The simulator uses the radar position and antenna attitude parameters to determine the radar ground footprint position and linearly interpolates the motion parameters for every sampling position in the flight direction. Currently, this computing task is carried out by the controller board of this raw signal simulator and then these parameters are distributed to every computing board.

Fig. 1. Raw data simulation block diagram

3 Processor design for raw data simulation

In order to satisfy the need for processing TD SAR raw signal simulation algorithms in real-time, in this paper, multiple FPGAs are used in parallel. Every computing FPGA is composed of many processors where the architecture and function of the processors are shown in Fig. 2. Details about the processor design will be described in detail in this section.

3.1 Memory architecture and parallel computing architecture

Because of the large SAR image area and enormous computing task, multiple FPGAs and multiple computing boards are needed. Accordingly, the RCS data for the whole simulated scene is distributed so that every FPGA can work in parallel without excess communication. There are basically two methods for data distribution: one is allocating large external memory covering the whole scene for every FPGA, the other is dividing the whole simulated scene into multiple FPGA memories. In this system we use DDRII-SRAM as the FPGA's external memory, which is unable to store the whole scene of a

single FPGA. As such, we chose the second method for data distribution. However, this method also works for SDRAM type memory where a larger simulation area is needed.

Fig. 2. Implementation architecture for raw signal generation

Fig. 3. Memory architecture of the simulator

The whole simulated scene is distributed to every FPGA memory and every FPGA should cover an almost equal size area for each radar pulse to maximise the parallel capability. According to the radar radiation pattern the area can be divided in the range or azimuth direction. For instance, in Fig. 3 the scene is divided into eight blocks in the range direction, which correspond with eight processing FPGAs. When the processing of signal coherent accu-

mulation is complete, the result of every processing unit needs to be accumulated for every processing module.

3.2 Slant range computing

Fig. 4. SAR system geometry

Accurate computing of the slant range and signal phase is essential for high quality SAR raw signal generation. The SAR system geometry is shown in Fig. 4, where $P(x, y, z)$ represents any point within the radar footprint and (X_0, Y_0, Z_0) represents the coordinate of the platform. The slant range and the radar echo signal phase are given by

$$R_s = \sqrt{(x - X_0)^2 + (y - Y_0)^2 + (z - Z_0)^2},$$

$$\Phi_s = \exp[-j4\pi R_s/\lambda] \quad (2)$$

For example, with the centre frequency at 10 GHz and a 3° phase error, the range error should be below 0.0012 m. If the slant range is computed using floating point method, double floating point square root is needed. The FPGA resource cost for floating point processing is heavy and latency is greatly increased with the precision. So in this paper, the square root is calculated in fixed point.

There are three kinds of square root methods which are commonly used: Newton-Raphson, SRT-redundant and non-restoring techniques. The Newton-Raphson method is an iterative approximation technique which requires multiple multipliers when a pipelined implementation is needed. The SRT-redundant is recursive and rather complex especially for high-radix SRT algorithm.

The use of a non-restoring square root algorithm is a simple and efficient technique because only addition/subtraction and shift operations are needed. In [5], a non-restoring square root algorithm was first proposed and much research focusing on optimising the non-restoring algorithm has been done. In [6], a modified non-restoring square root algorithm is presented. Its main difference is to use a subtract operation and append a binary value of 01 instead of appending 11 and adding to the developed root. In [7], a modular non-restoring square root algorithm is proposed. Several elements are removed without accuracy loss, while achieving significant reduction in the required area. Results show that the hardware resources usage is reduced.

Although the method proposed in [6] is characterised by a significant improvement in resource utilisation, the register-transfer level (RTL) implementation of this method is not suitable because of much higher resource cost. Besides, the processors in [6] and [7] are only implemented in non-pipelined FPGA structure with no clock or flip flops.

In this paper, a modified non-restoring pipelined architecture is used to optimise hardware resources by taking advantage of the FPGA's internal CLB structure especially optimized for adder realization and using the RTL design directly. In every pipelined stage of the non-restoring algorithm the adder and subtractor are multiplexed using one complement adder. The hardware resource report of the 64 bit input square root processors is shown in Table I according to the Xilinx ISE12.4 using FPGA XC4VSX55. It can be seen that the LUTs decrease when comparing the processors in [6, 7].

Table I. Resource report

	Proposed	Samavi [7]	Sutikno [6]
Slice Flip Flops	1150		
Slice LUTs	648	1073	1899
Slices	637	617	987

The phase accuracy of the raw signal is directly related to the slant range accuracy and the fixed point length of the square root. As an example, in this paper the phase error is less than 3 degrees and the fixed point length of the data is determined as 30 bits.

3.3 Phase computing

The signal phase is calculated from the slant range and the complex exponential is generated using the look-up table. According to the time based SAR raw signal algorithm the range gate for every point and the phase remainder need to be calculated. In order to preserve the phase accuracy, a 30 bit fixed point multiplier is used in Eq. (3), where the *Rem_phase* is used as the address for the cosine-sine look-up table, *Quan_width* is equal to the bit width of the slant range computing, *Phase_width* is the bit width of the sine look-up table, and $[2^{Range_width+1} \cdot 2^{Phase_width}] / \lambda$ can be calculated and stored as a constant.

$$Rem_phase = [2^{Quan_length+1} \cdot 2^{Phase_length}] / \lambda \cdot R_s / 2^{Quan_length} \quad (3)$$

The raw signal phase error between the double floating point and the fixed point is shown in Fig. 5. The radar parameters are listed in Table IV and slant ranges are 18 km, 20 km and 22 km respectively.

The complex exponential of signal phase result is then multiplied by the RCS parameters and then stored in the FPGA internal Block RAM for coherent accumulation.

3.4 Radar return signal coherent accumulation

Radar return signal accumulation for every range gate is designed to optimise

Fig. 5. Cosine and sine error of the signal phase

the memory resource. The accumulation is carried out using the dual port memory embedded in the new generation FPGA, where the data can be read and written simultaneously. The address of memory is increased according to the slant range and a delayed write process is proposed so as to avoid the read and write conflict with the same address. After the accumulation for all processors is completed, the result is read out sequentially according to the slant range. In order to increase processing speed, swinging or ping-pong memory is used.

3.5 Convolution

After the accumulation processing step, the processed data needs to be convoluted with the transmitted radar signal to generate the raw signal. The convolution can be performed using the direct time domain convolution or the fast convolution method using FFT. The former needs great resource but has less latency when the operation can be performed in parallel. The resource of the latter is reduced however the latency for the first output is increased. In this paper we use the pipelined-FFT processor to deal with this problem. Among all kinds of pipelined FFT processor, the R2²SDF [8] architecture is selected for the high speed and medium resource cost which can process in N clock period for N points FFT. The drawback of the R2²SDF is the output order is bit-reversed. If we use the same structure for the FFT and IFFT there needs to be additional memory to reorder the output result of FFT and the latency is greatly increased. So the DIT and DIF structures are presented for the FFT and IFFT processor respectively. In this simulator the 2048 points FFT and IFFT processor are used and the structures are shown in Fig. 6 and Fig. 7.

Fig. 6. R2²SDF DIF structure

Fig. 7. R^2SDF DIT structure

4 System implementation and performance

4.1 System architecture

Fig. 8. The raw signal simulator system architecture

As shown in Fig. 8, the simulator is founded on compact PCI architecture with five computing modules, one controller module and one analog signal generating module. The controller board is an Intel single board computer CT11 from GE Fanuc. It is used to control the workflow of this simulator and is responsible for communication with other devices to get platform motion parameters through reflection memory interface and backscattering coefficient data through Ethernet interface. As described in section 2, the controller board is also used to computing the radar position and ground footprint position. These parameters are then downloaded to analog signal generating board and distributed to every computing board. The simulator controlling communication is based on PCI bus and the high speed data transferring between the modules is through backplane LVDS bus.

The hardware structure and implementation of the processing modules are shown in Fig. 9 which are designed and implemented by the authors. Each computing module consists of five Xilinx's XC5VSX95T which are responsible for raw signal computing and we use DDRII-SRAM as external memory because of its high speed, low delay capability and relatively simplified controller interface.

The analog signal generating module is composed of one XC5VSX95T and one XC5VSX55T. The FPGA XC5VSX95T is used to receive the digital raw signal and converts to analog base-band signal through DAC AD9736. The FPGA XC4VSX55 is used to transmit simulating parameters to computing module.

Fig. 9. The hardware structure of processing modules

Fig. 10. The hardware integration of simulator

The system integration is shown in Fig. 10, in which there are two computing modules for initial test.

4.2 FPGA performance

As previously described, there are mainly two kinds of FPGA modules in the system; one is computing module and the other is analog signal generating module. We designed the FPGA according to the radar parameters list in Table IV.

4.2.1 Computing board FPGA

On the computing board there are four FPGAs for main computing and one

Table II. The main computing FPGA resource cost

Logic Utilisation	Used	Available	Utilisation
Number of Slice Registers	39,423	58880	66%
Number of Slice LUTs	41,132	58880	69%
Number of occupied Slices	14,245	14,720	96%
Number of Block RAM	173	244	70%

FPGA for data accumulation and communication. Every main computing FPGA has 32 processors which can carry out pipelining tasks including range, phase computing and signal accumulation.

We use the Xilinx ISE12.4 for synthesis, place and route. The cost of the main computing FPGA is listed in Table II and the synthesis speed for main clock can reach 115 MHz.

4.2.2 Analog signal generating board FPGA

On the analog signal generating board, there is one XC5VSX95T for convolution and analog signal generation, and one XC4VSX55 for communication. The resource cost of the XC5VSX95T is list in Table III.

Table III. The analog signal generating FPGA resource cost

Logic Utilisation	Used	Available	Utilisation
Number of Slice Registers	10,169	58880	17%
Number of Slice LUTs	8,828	58880	14%
Number of occupied Slices	3,569	14,720	24%
Number of Block RAM	40	244	16%

Every PRF, the radar parameters are transferred to every computing module and after the raw signal computing is completed in the FPGA, the data is converted to analog signal.

The system is implemented and tested with the parameters list in Table IV. The analog raw signal waveform is shown in Fig. 11. It shows that the continuous computing time is below one PRT (pulse repetition time, 25 μ s) and needs less than 51.2 ms for 2048 pulses simulation.

As an example, we use the backscattering coefficient shown in Fig. 12(a) for testing and the radar parameters is list in Table IV. In order to illustrate the radar motion parameters impact on SAR imaging, the simulation is carried out with two different trajectories. The first trajectory is an ideal line with constant velocity, the second trajectory is added a random deviation of amplitude of 2.5 meters in x/y/z direction and the parameters update rate is 10 ms. The simulator then generates the raw signal according to the platform and radar parameters. The acquisition board receives the radar raw signal and focuses the SAR image which is shown in Fig. 12(b) and Fig. 12(c).

Fig. 11. Process diagram of the simulator

Table IV. SAR simulation parameters

Frequency (GHz)	10
PRF (Hz)	40000
Slant range of scene centre (km)	20
Azimuth resolution (m)	0.3
Range resolution (m)	3
Squint angle (deg)	0
Azimuth points	640
Range points	640
Bandwidth (MHz)	60

Fig. 12. The SAR imaging result of the raw signal data
(a) Original image of backscattering coefficient (b) Focused image of the raw signal data with ideal platform trajectory (c) Focused image of the raw signal data with random deviation from ideal platform trajectory

5 Conclusion

In this paper an FPGA-based system for real time SAR raw signal generation is presented and it can be used in HWIL simulation. The simulator is based on the TD algorithm and the optimisation is presented to decrease the FPGA resource cost. The slant range computing is based on a fixed point modified

pipelined non-restoring algorithm which take advantages the FPGA internal structure. It can achieve a phase error of less than three degrees and has a lower resource cost when compared with the existing methods. The memory architecture of the simulated scene is designed to parallelise all computing nodes and the simultaneous write/read BRAM are used to decrease the memory usage for signal accumulation. The convolution with the transmit signal is based on pipelined FFT/IFFT processors with DIT/DIF decomposition methods. The simulator is designed and implemented on a switch-based architecture with the advantage of flexible extension. The system is implemented and verified with test results which can simulate a scene size of $640 * 640$ point scatters with PRF higher than 40 kHz.