

On-Board Fault-Tolerant SAR Processor for Spaceborne Imaging Radar Systems

Wai-Chi Fang , Charles Le, and Stephanie Taft

Jet Propulsion Laboratory, California Institute of Technology

4800 Oak Grove Drive, Pasadena, CA 91109-8099 USA

***Abstract-* A real-time high-performance and fault-tolerant FPGA-based hardware architecture for the processing of synthetic aperture radar (SAR) images has been developed for advanced spaceborne radar imaging systems. In this paper, we present the integrated design approach, from top-level algorithm specifications, system architectures, design methodology, functional verification, performance validation, down to hardware design and implementation.**

I. INTRODUCTION

In anticipation of many potential joint Space-Based-Radar (SBR) missions between the Air Force and NASA in the future [1], we propose to develop an FPGA-based (field-programmable-gated-array) architecture for onboard processing of radar data. In particular, the hardware is targeted for the high computational load in processing synthetic-aperture-radar (SAR) and space-time-adaptive-processing (STAP) algorithms. The real-time processing capability of such algorithms has been identified by NASA and the Air Force as the needed technology, enabling fast turn-around and direct distributions of relevant information to a number of potential users. For commercial applications, the availability of SAR images can be processed further in the post-processor to deliver final value-added products directly to users. For governmental uses, real-time SAR images and additional information derived from them (polarimetric, DTED) can render the algorithms performed by other sensors more robust (knowledge-based processing). Hence, development and demonstration of a real-time SAR processor could lower one of the major technological risks in future spaceborne SAR systems. Such an onboard processor, however, faces many conflicting design challenges that are usually demanded at the same time. For example, some of the major objectives are:

- To process, in real-time, SAR and STAP algorithms in the specified radar modes, with a required performance and accuracy.
- To exhibit reasonable programmability and reconfigurability for possible algorithmic updates after launch.
- To have adequate scalability with respect to changes in system parameters to sustain 2x in computation and communication.
- To demonstrate a best-practice testability.
- To satisfy the stated latency and update rate.
- To interface within the radar electronic system.
- To fit into the requisite physical constraints
- To have a minimum mission lifetime of three years with graceful degradation.

- To be tolerant to the space radiation environment at the specified orbital altitude, with a given reliability and availability.
- To be realizable and low-risk
- To have an open architecture and a clearly defined technology upgrade path.

It was found that the most suitable architecture which satisfies these requirements and constraints while providing enough flexibility would be a *hybrid system*. The computational unit consists of a *FPGA-based front-end* for fixed-point regular operations (e.g. I/Q demodulation, equalization, conventional or adaptive digital beamforming, pulse compression, and possibly Doppler processing or image formation), coupled with a *programmable processor-based back-end* for adaptive floating-point algorithms (e.g. image formation, space time-adaptive processing, target classification). The Air Force Research Laboratory in Rome, NY is focusing its work on the back-end portion, with leverage from its previously developed Wafer Scale Signal Processor [2]. The Jet Propulsion Laboratory devotes its effort on the FPGA front-end, which is the topic of this paper. In the subsequent sections, we will discuss the integrated and systematic approach undertaken to design a high-performance and fault-tolerant FPGA-based processor for SAR processing.

II. ALGORITHM SPECIFICATIONS AND SYSTEM REQUIREMENTS

The SAR processing algorithm assumes the well-studied Range-Doppler image formation technique [3].

The general block diagram of the Range-Doppler SAR processor is shown in Fig. 1. The processing steps are shown in Fig.2. The input processor, which is part of the Digital Beamforming Processor, takes in 12-bit offset-video data and produces 16-bit I/Q baseband data. Pulse compression is performed in the range processor using FFT. The data are then corner-turned, followed by azimuth compression with range migration in the Doppler frequency domain (an optional spotlight-mode processing is reserved for future study). Finally, data is power-detected, averaged over a specified number of looks, and formatted for storage or down-link. Two SAR modes have been identified for real-time demonstration: high-resolution and wide-swath modes. The relevant radar parameters are described in Table 1.

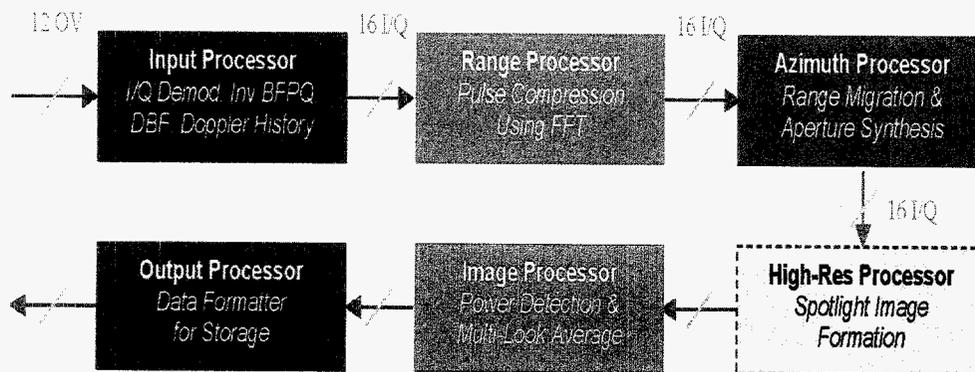


Fig. 1: SAR processing data flow.

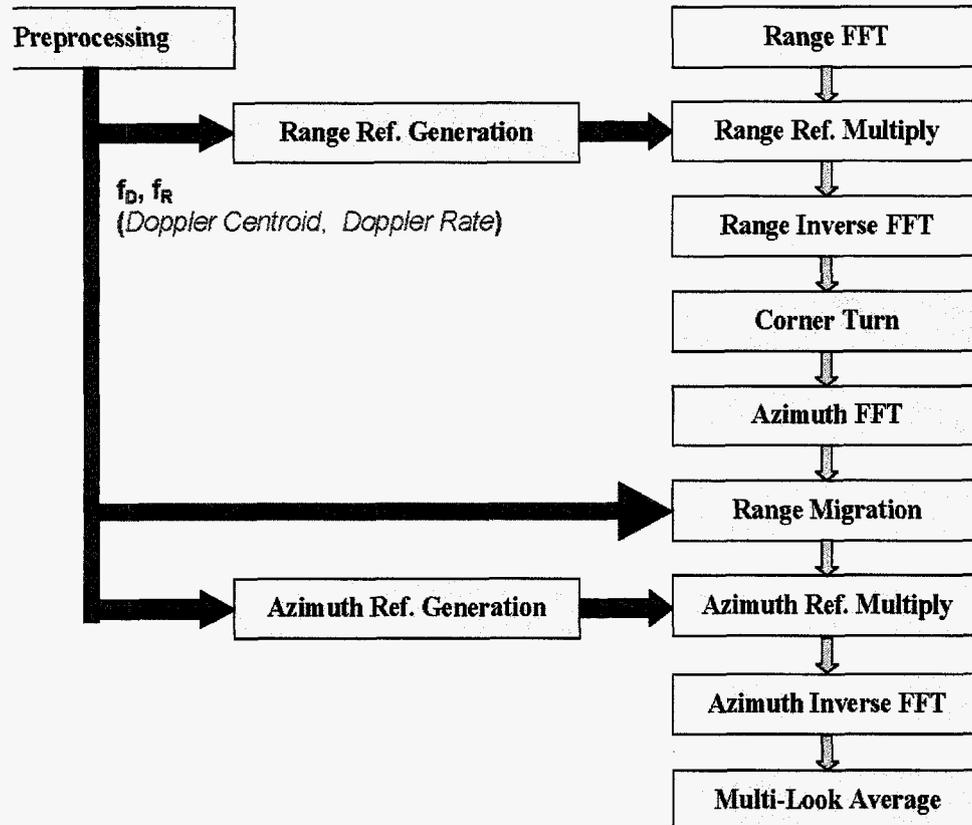


Fig. 2: SAR processing steps.

TABLE 1 Radar Parameters for SAR Modes

Parameters	High-Resolution SAR	Wide-Swath SAR
Antenna Width, m	3	0.35
Antenna Length, m	10	50
Operating Frequency, MHz	1260	1260
Pulse Width, μ sec	10	100
Radar Bandwidth, MHz	65	45
Pulse Repetition Frequency, Hz	1782	350
Orbit Altitude, km	500	500
Incident Angle, degree	33	33
Data Sampling Window, μ sec	73	907
Total Throughput, GFLOPS	3.44	18
Aggregate Data Rate, Gbits/s	1.02	1.78

III. DESIGN METHODOLOGY

Hardware design usually consists of two major tasks: a) system/algorithm design team specifies the system requirements, develops floating-point models, performs mathematical analysis and simulation, and produces design specifications, which are given to b) hardware/software implementation team to interpret and map them to H/W and S/W implementation. Very often, the two teams use different approaches (mathematical complexity for algorithm developers versus data flow for hardware designers), design languages (C/C++ versus HDL), and different toolsets (interactive numerical packages versus circuit simulators). This is a classical hardware design problem that is error-prone, leading to many product re-design cycles. Hence, there is a strong need for an integrated design methodology where the above disparity is removed by having a common development platform used by both teams. This environment would allow the entire team to model, simulate, and validate, in a modular structure and throughout the whole design process, the design at multiple levels of abstraction, and consequently verify that the design will function properly when first implemented. For the above reasons, the team has adopted the design approach proposed by Cadence's Signal Processing Work system (SPW) [5].

IV. SIMULATION, VERIFICATION, AND DEMONSTRATION

A. Simulation

The simulation and verification plan is shown in Fig. 3. First, the test vectors comprise of the point target signal, corresponding to the relevant mission parameters, and SIR-C SAR data. The purpose of the point target signal is to measure the radar performance parameters such as the peak sidelobe ratio (PSLR), the integrated sidelobe ratio (ISLR), the resolution broadening, and the phase fidelity. SIR-C data is used to assess the performance according to some chosen scientific measures. One such parameter is the radiometric accuracy which is related to the dynamic range of the hardware implemented in fixed-point format.

In the Algorithm Simulator, a "golden model" (executable specification) is written in a C language, using floating-point format to implement the SAR processing algorithms. It will process the point target signal from the point target simulator, and SIR-C data. Its outputs serve as the standard against which the hardware will be assessed. The floating-point model has the option of outputting the data at various intermediate stages for the verification purpose of individual hardware module. A test bench will be written so that portions (e.g. the range compression module) of the floating-point model can be executed in a hardware prototyping circuit. Usually, the ISLR, PSLR, resolution, and phase are measured here to verify the floating-point performance of the algorithm. Next comes the HDL simulator where the algorithm is captured in floating-point format and converted

into fixed-point representation. A hardware design and development tool (SPW) will translate its block diagram into a HDL representation of the algorithm. Frequently, simulation is done at this point to assess the radiometric and phase accuracy, and hence verifying the correctness of the parallel and pipeline design, before mapping to hardware. Finally, in the Circuit Emulator, a hardware architecture (e.g. Xilinx's Virtex-II Pro) is chosen to synthesize the HDL representation into circuit logics, followed by place-and-route on a chip. The hardware will process the signal in real time. Functional performance, such as timing, power, size, weight, latency, scalability, etc. is commonly performed here.

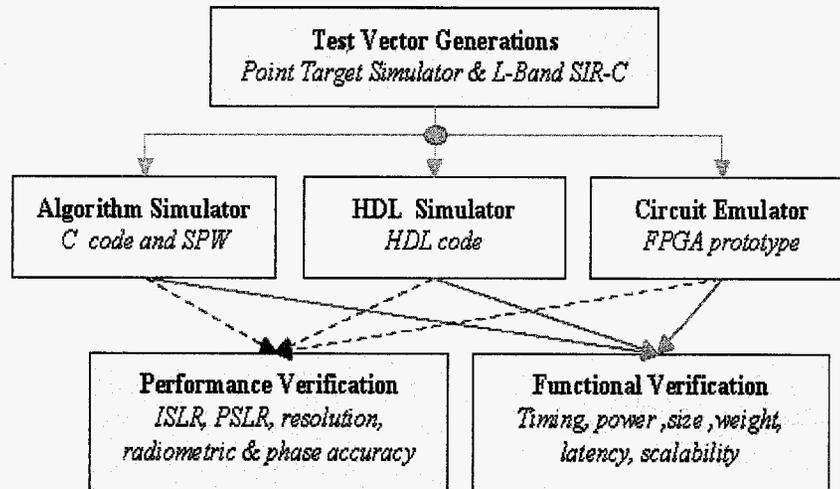


Fig. 3: Simulation and verification plan.

B. Performance and Functional Verification

Since the hardware is implemented using fixed-point arithmetic, the effects of its limited wordsize on various radar performance parameters are measured and quantified. The fixed-point SAR model will be compared with the floating-point model. Similarly, functional verification is done at each sub-module (range compression, range migration, azimuth compression) to verify its functional correctness, before data is processed in the next modules. The fixed-point and HDL SAR models are simulated and compared to assure bit-by-bit match. The required functional and performance parameters are summarized in Table 2.

TABLE 2 Verification Parameters

Performance Parameter	Value
ISLR, dB	-13
PSLR, dB	-20
Resolution Broadening	1.05
Radiometric Accuracy, dB	0.5
Phase Fidelity, degree	1

Functional Parameter	Value
Latency, sec	5
Power, watt	100
Size, cm ³	20x20x20
Weight, kg	5
Scalability	2X
Reliability	0.99999

V. FAULT-TOLERANT STRATEGY

In order to survive the space radiation environment, a highly reliable processor is needed for space-based radars and other planetary explorations (e.g. Jupiter). However, available radiation hardened components usually fall a few generations behind commercial products, create trade-off in throughput and power consumption, have low-market demand, are scarcely available and quickly outpaced by technology change, and increase mission cost. Very often, fault-tolerant circuit design can be effectively employed by using a hybrid approach of integrated fault-mitigation technologies (at the system, component, or logics levels combined with time, software, and information redundancy) to leverage commercial rad-tolerant devices. Such electronic circuits are required to be able to survive in space environments (temperature, radiation) and to have long operational lifetimes (> 10 years) with graceful degradation thereafter. It should be operational even in the presence of faults.

To achieve those objectives, a dynamic fault-tolerant on-board processor is recommended as shown in Fig. 4. In this first iteration, triple modular redundancy (TMR) is implemented at the system level (the whole SAR processing chain). Fault-tolerant strategy is implemented in the Fault-Management Unit which consists of four main functions

- the **Scrub Controller** which executes periodic reloads of FPGA configuration data for SEU correction,
- the **Fault Detection Circuit** which detects the faulted processor by executing periodic tests (periodic tests temporarily suspend the processor's normal operations and a test routine is run to determine if faults are present in the processor),
- the **Switching Circuit** which removes the faulty processor from normal service and selects the system output to come from one of the alternative processors,
- and the **Majority Voter Circuit** which implements the majority voting function by taking the majority voted output, at the output clock rate, from three identical processors.

Subsequent design iterations will consider redundancy at the next abstraction levels (circuit, logic, and gate levels). With this approach, the computational resource needed is at least triple, excluding voting and additional supporting circuitry. To reduce the real estate, other advanced fault-tolerant strategies could be equally applied, such as algorithm-based fault tolerance, software redundancy, time redundancy, and information redundancy.

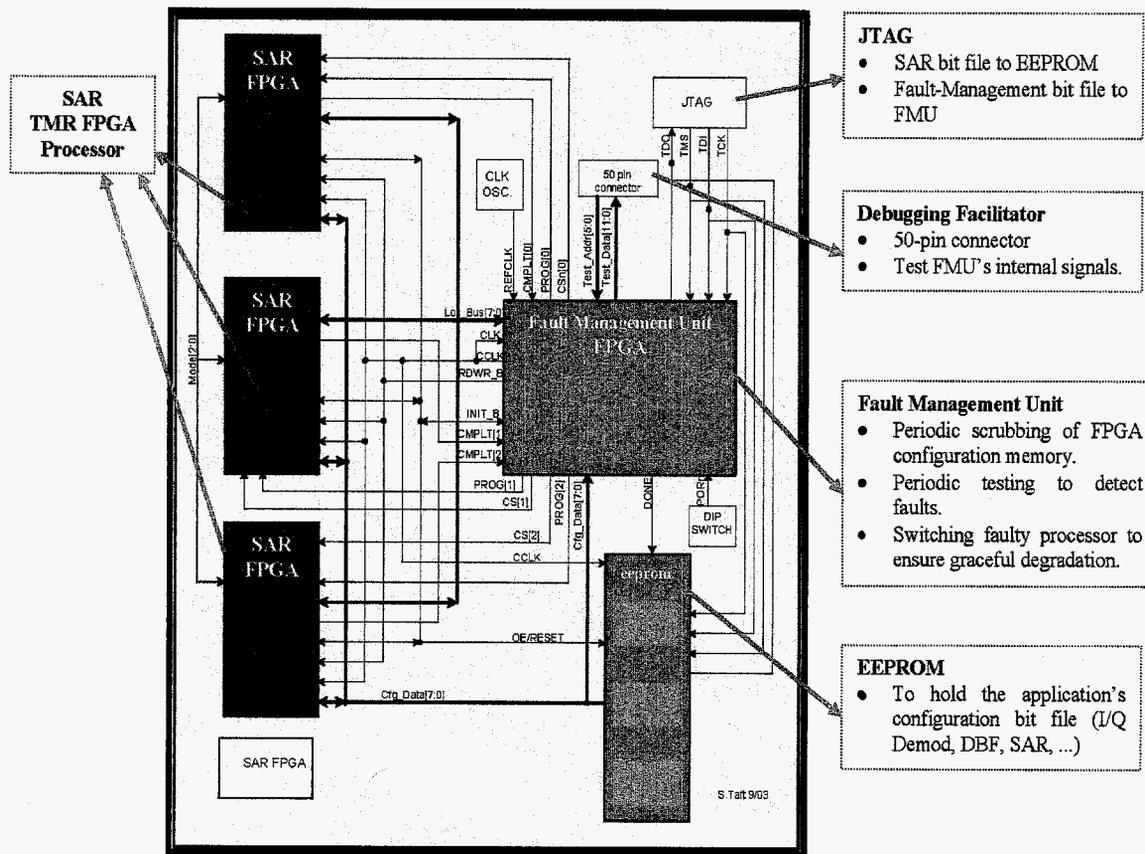


Figure 4: Dynamic TMR fault-tolerant on-board processor with Fault Management Unit.

VIII. CONCLUSION

A high-performance and fault-tolerant FPGA-based hardware architecture for the processing of SAR images in future spaceborne radars was presented, together with the system requirements and algorithm specifications, design methodology, simulation and verification methods, testing and demonstration plans, interface circuit design, and fault-tolerant strategies. Preliminary test by executing range compression in hardware produces promising results.

ACKNOWLEDGMENT

The authors wish to thank members of the SBR team for fruitful discussions and many valuable suggestions. This work was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

REFERENCES

- [1] P. Rosen and M. Davis, "A joint space-borne radar technology demonstration mission for NASA and the Air Force," *Proc. IEEE 2003 Aerospace Conf.*, Big Sky MT, vol. 1, pp. 437-444, March 2003.
- [2] R.W. Linderman, R.L.R. Kohler, and M.H. Linderman, *IEEE Trans. Computers*, 3rd ed., vol. 47, no. 1, pp. 125-128, January 1998.
- [3] J.C. Curlander and R.N. McDonough, *Synthetic Aperture Radar System and Signal Processing*, Wiley Interscience, NY, 1991
- [4] "FPGA Design with Cadence SPW", Datasheet, Cadence Design Systems, Inc, 2002.