Onboard FPGA-Based SAR Processing for Future Spaceborne Systems

Charles Le, Samuel Chan, Frank Cheng, Winston Fang, Mark Fischman, Scott Hensley, Robert Johnson, Michael Jourdan, Miguel Marina, Bruce Parham, Francois Rogez, Paul Rosen, Biren Shah, and Stephanie Taft
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109-8099 USA

Abstract- **We present a real-time high-performance and fault-tolerant FPGA-based hardware architecture for the processing of synthetic aperture radar (SAR) images in future spaceborne system. In particular, we will discuss the integrated design approach, from top-level algorithm specifications and system requirements, design methodology, functional verification and 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 (range-Doppler) and STAP (Joint-Domain Localized) 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 testability according to best practice.
- To satisfy the stated latency and update rate.
- To interface smoothly within the radar electronic system.
- To fit into the requisite physical constraints (size, weight, and power).
- 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 (in terms of performance, schedule, cost, and on-orbit maintainability).
- 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. inverse block floating-point quantization (Inv BFPQ), I/Q demodulation, equalization, conventional or adaptive digital beamforming (DBF), 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 real time high-performance and fault-tolerant FPGA-based hardware architecture for SAR processing.

II. Algorithm Specifications and System Requirements

The SAR processing algorithm assumes the well-studied Range-Doppler image formation technique [3, 4]. Range-Doppler is better suited to high-frequency radars that may have considerable Doppler variation across the swath. In this first development of a hardware implementation for real-time SAR processing on a stable satellite platform, motion perturbation is assumed to be negligible. The demonstration on a hardware prototype will be performed on simulated point target [5] and L-band SIR-C data [6].

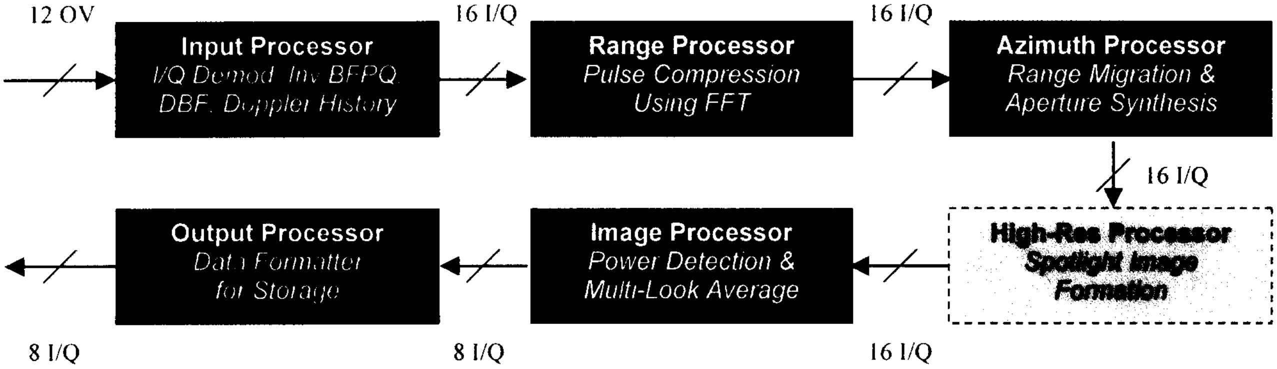


Fig. 1: SAR processing data flow.

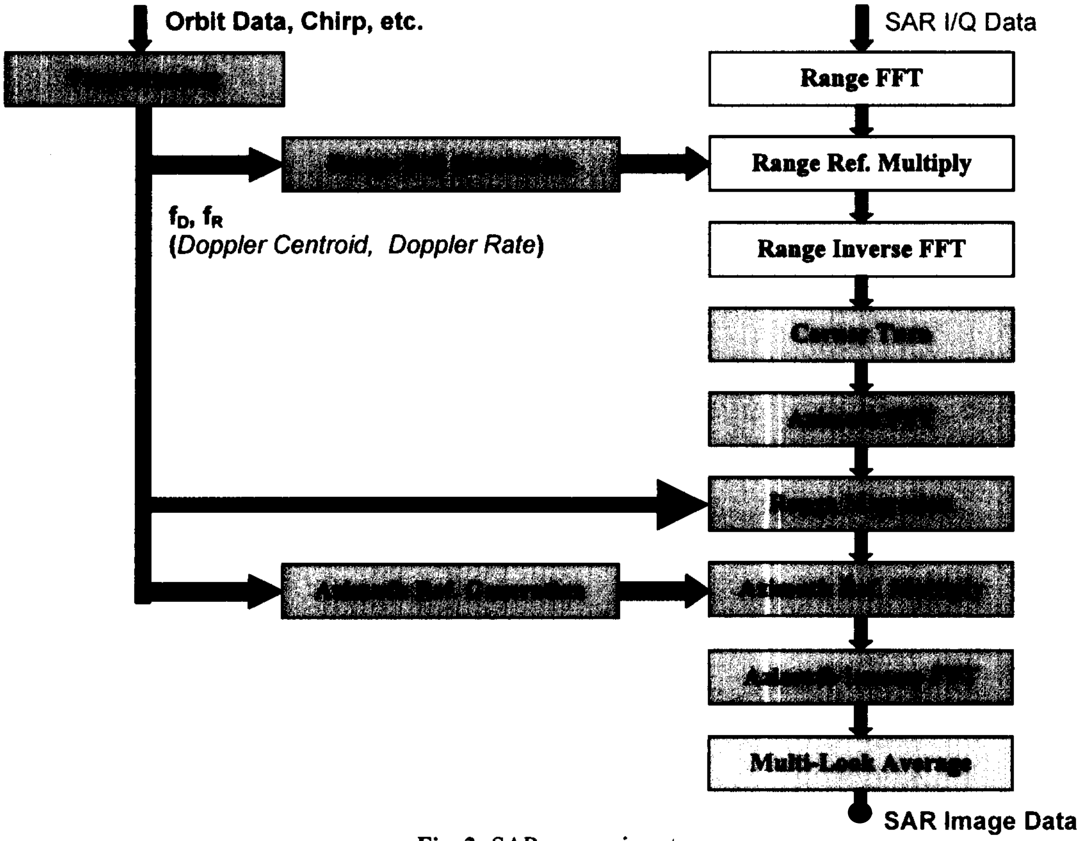


Fig. 2: SAR processing steps.

The general block diagram of the Range-Doppler SAR processor is shown in Fig. 1. 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. The processing steps are shown in Fig. 2 (refer to [7] for a detailed description of the algorithm).

During the last two years, the Air Force and NASA have been working together to define a joint SBR Demonstration Mission to be flown in the year 2008 [1]. Two SAR modes have been identified for real-time demonstration: high-resolution and wide-swath modes. The relevant radar parameters are described in the tables below.

TABLE 1
HIGH-RESOLUTION SAR MODE

Parameters	Value
Antenna Width, m	3
Antenna Length, m	10
Operating Frequency, MHz	1260
Pulse Width, μsec	10
Radar Bandwidth, MHz	65
Pulse Repetition Frequency, Hz	1782
Orbit Altitude, km	500
Incident Angle, degree	33
Data Sampling Window, μsec	73
Total Throughput, GFLOPS	3.44
Aggregate Data Rate, Gbits/s	1.02

TABLE 2
WIDE-SWATH SAR MODE

Parameters	Value
Antenna Width, m	0.35
Antenna Length, m	50
Operating Frequency, MHz	1260
Pulse Width, μsec	100
Radar Bandwidth, MHz	45
Pulse Repetition Frequency, Hz	350
Orbit Altitude, km	500
Incident Angle, degree	33
Data Sampling Window, μsec	907
Total Throughput, GFLOPS	18
Aggregate Data Rate, Gbits/s	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 Worksystem (SPW), whose methodology is shown in Fig. 3 [8].

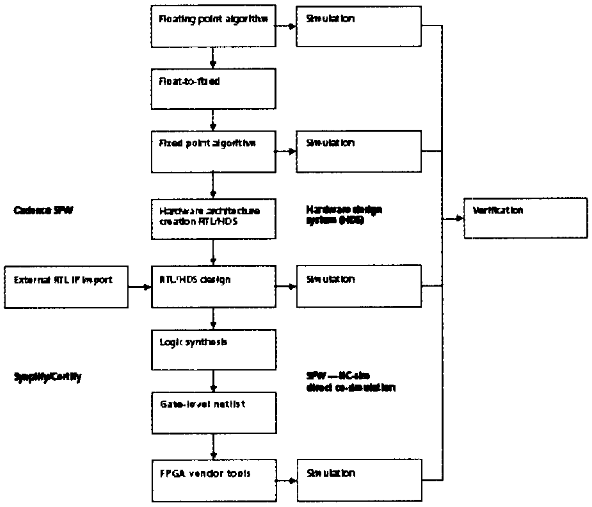


Fig. 3: SPW-based FPGA design methodology.

IV. Simulation, Verification, and Demonstration

A. Simulation

The simulation and verification plan is shown in Fig. 4. First, the test vectors contain 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.

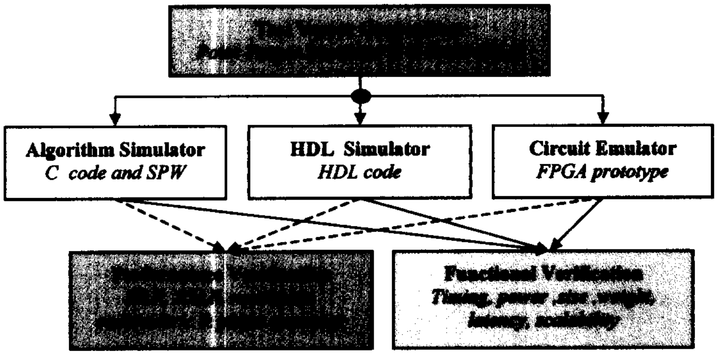


Fig. 4: Simulation and verification plan.

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 writing out 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.

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. The required radar performance parameters are summarized in Table 3.

TABLE 3
PERFORMANCE PARAMETERS

Parameter	Value
ISLR (one dimension), dB	-13
PSLR, dB	-20
Resolution Broadening	1.05
Radiometric Accuracy, dB	0.5
Phase Fidelity, degree	1

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 parameters are summarized in Table 4.

TABLE 4
FUNCTIONAL PARAMETERS

Parameter	Value
Latency, sec	5
Power, watt	100
Size, cm3	20x20x20
Weight, kg	5
Scalability	2X
Reliability	0.99999

C. Demonstration

A demonstration plan targeting for Apr. 2004 and incorporating both the algorithm simulator and the circuit emulator is shown in Fig. 5. Here, the input raw data (simulated point target signal or SIR-C data) are range-compressed in an FPGA-prototype circuit in real time. The rest of the SAR processing chain will be processed in the software golden model (ROIPAC.c) or simulated using SPW's fixed-point model (ROIPAC.v). The results are then compared with those obtained by using the software model alone, for functional and performance verification.

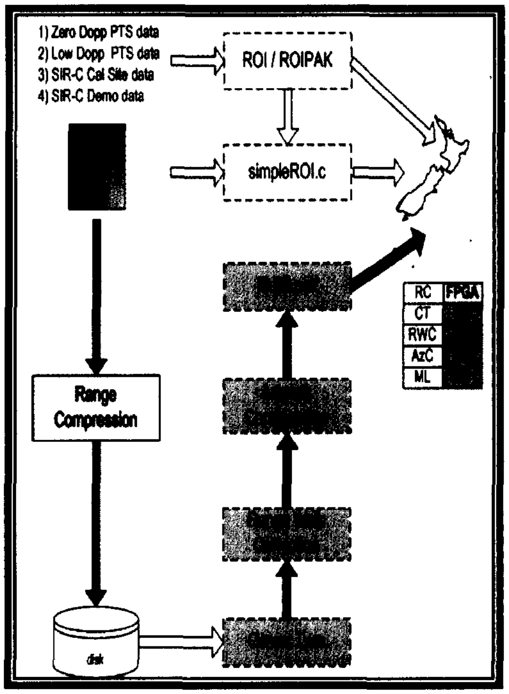
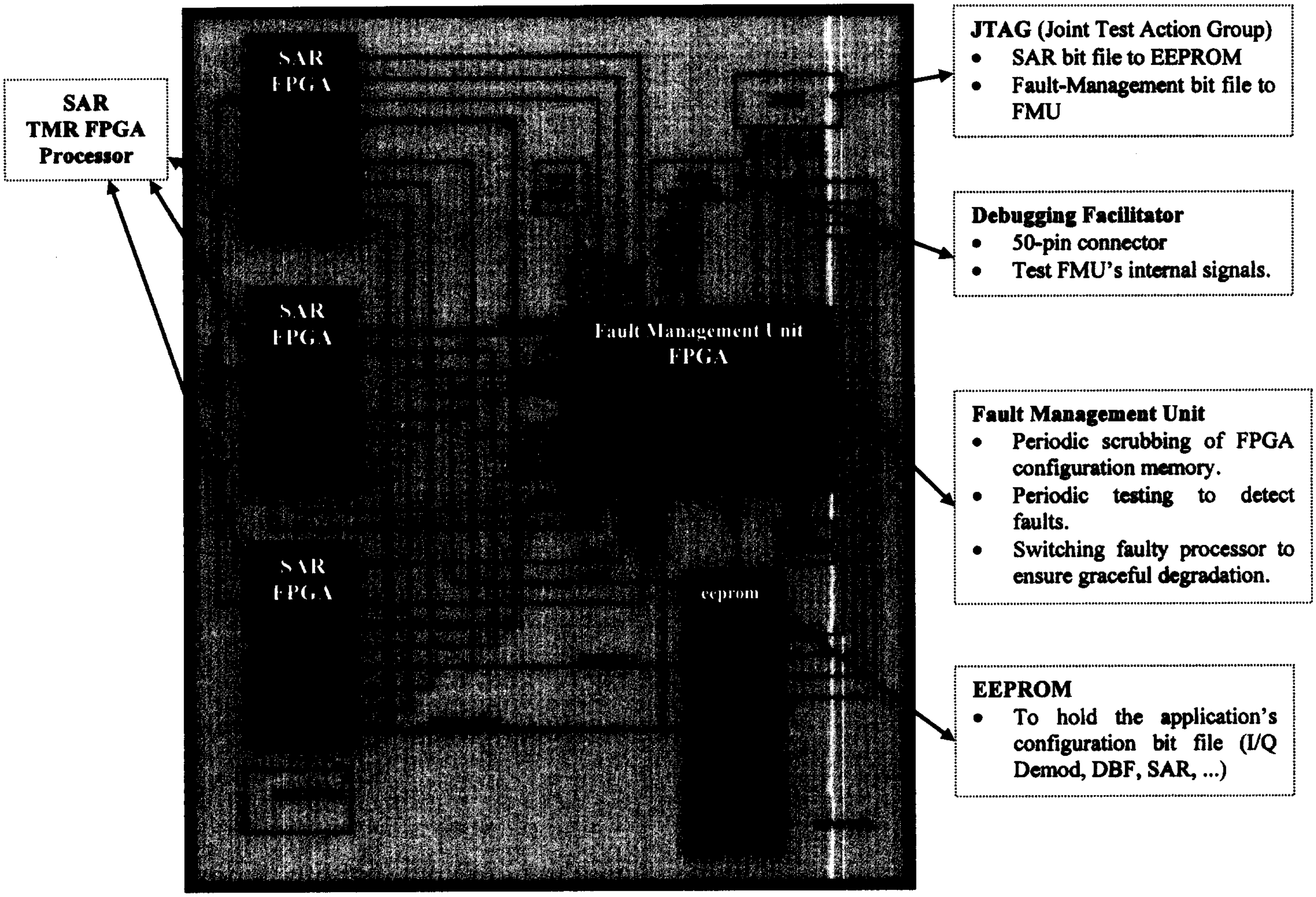


Fig. 5: Demonstration plan.

V. FAULT-TOLERANT STRATEGY

In order to survive the space radiation environment, a highly reliable processor is needed for space-based radar and other planetary exploration (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 present of faults.

To achieve those objectives, a dynamic fault-tolerant on-board processor is recommended as shown in Fig. 6. In this first iteration, triple modular redundancy (TMR) is



SAR TMR FPGA Processor

JTAG (Joint Test Action Group)
- **SAR bit file to EEPROM**
- **Fault-Management bit file to FMU**

Debugging Facilitator
- **50-pin connector**
- **Test FMU's internal signals.**

Fault Management Unit
- **Periodic scrubbing of FPGA configuration memory.**
- **Periodic testing to detect faults.**
- **Switching faulty processor to ensure graceful degradation.**

EEPROM
- **To hold the application's configuration bit file (I/Q Demod, DBF, SAR, ...)**

Figure 6: Dynamic TMR fault-tolerant on-board processor with Fault Management Unit.

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. These are the topics of current research.

VI. Interface Specification

The interface between the antenna panel, the front-end receiver module, the digital receiver, the digital beamformer, the SAR processing block, and the solid-state recorder is displayed in Fig. 7. Note that this interface is designed to be general and flexible enough so that its parameters could be easily modified to accommodate new system requirements.

VII. Preliminary Results

Preliminary results using the simulated point target signal are

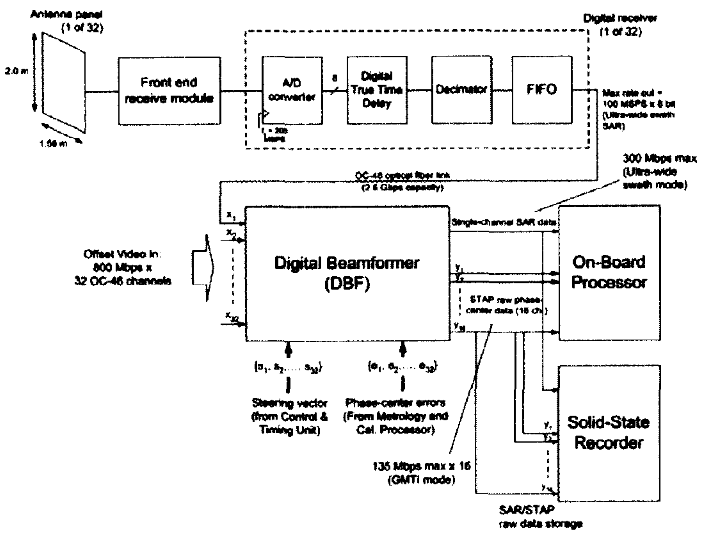

Antenna panel (1 of 32)
2.0 m
1.56 m
Front end receive module
Digital receiver (1 of 32)
A/D converter
8
f_s = 200 Msps
Digital True Time Delay
Decimator
FIFO
Max rate out = 100 MSPS x 8 bit (Ultra-wide swath SAR)
300 Mbps max (Ultra-wide swath mode)
OC-48 optical fiber link (2.5 Gbps capacity)
Offset Video In: 800 Mbps x 32 OC-48 channels
x_1
x_2
x_{32}
Digital Beamformer (DBF)
Single-channel SAR data
y_1
y_2
STAP raw phase-center data (16 ch)
y_{16}
On-Board Processor
Solid-State Recorder
$\{s_1, s_2, \ldots, s_{32}\}$
$\{e_1, e_2, \ldots, e_{32}\}$
Steering vector (from Control & Timing Unit)
Phase-center errors (From Metrology and Cal. Processor)
135 Mbps max x 16 (GMTI mode)
SAR/STAP raw data storage

Fig. 6: SBR interface.

shown in Figs. 7a (floating-point range compression), 7b (fixed-point range compression), and 7c (hardware range compression).

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 Dr. Mark Davis, Capt. Jules Bergmann, John Rooks, Ralph Kohler, and Dr. Richard Linderman of AFRL and Dr. Elaine Chapin of JPL 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*, 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] R. Bamler, "A comparison of range-Doppler and wavenumber domain SAR focusing algorithm", *IEEE Trans. Geosci. Remote Sensing*, vol. 30, no. 4, pp. 706-713, July 1992.

[5] S. Hensley, E. Chapin, A. Freedman, E. Gurrola, P. Kroger, C. Le, T. Michel, C. Shaffer, and C. Werner, *Point Target Simulator: User's Manual and Brief Algorithm Description*, Project Report, Jet Propulsion Laboratory, September 1999.

[6] R.L. Jordan, B.L. Huneycutt, and M. Werner, "The SIR-C/X-SAR synthetic aperture radar system", *IEEE Trans. Geosci. Remote Sensing*, vol. 33, no. 4, pp. 829-839, July 1995.

[7] S. Buckley, P. Rosen and P. Persaud, *ROIPAC Documentation*, Project Report, Jet Propulsion Laboratory, May 2000.

[8] "*FPGA Design with Cadence SPW*", Datasheet, Cadence Design Systems, Inc, 2002.

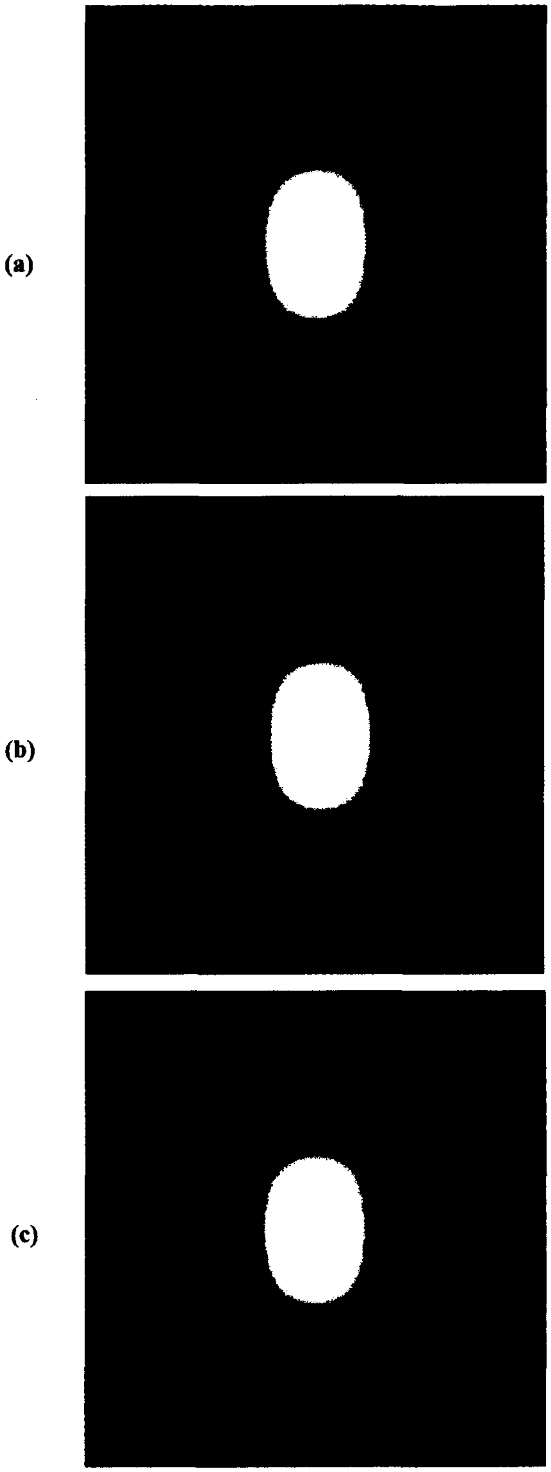


Fig. 7: Point target responses.