An FPGA-Based Doppler Processor for a Spaceborne Precipitation Radar

S. L. DURDEN, M. A. FISCHMAN, R. A. JOHNSON, A. J. CHU, M. N. JOURDAN, AND S. TANELLI

Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California

(Manuscript received 11 August 2006, in final form 21 February 2007)

ABSTRACT

Measurement of precipitation Doppler velocity by spaceborne radar is complicated by the large velocity of the satellite platform. Even if successive pulses are well correlated, the velocity measurement may be biased if the precipitation target does not uniformly fill the radar footprint. It has been previously shown that the bias in such situations can be reduced if full spectral processing is used. The authors present a processor based on field-programmable gate array (FPGA) technology that can be used for spectral processing of data acquired by future spaceborne precipitation radars. The requirements for and design of the Doppler processor are addressed. Simulation and laboratory test results show that the processor can meet real-time constraints while easily fitting in a single FPGA.

1. Introduction

Doppler velocity measurements from ground-based and airborne radars have played a key role in detecting and monitoring severe weather (Doviak and Zrnic 1993). Doppler measurements from space are complicated by the large platform velocity (Meneghini and Kozu 1990; Amayenc 1993) and have not yet been attempted. Tanelli et al. (2002) have shown that if rain within the radar beam is nonuniform then biases in the estimated Doppler velocity can result with the standard pulse-pair algorithm. One approach to accurate Doppler measurement from space is the combined frequency–time (CFT) technique (Tanelli et al. 2004), which requires the full Doppler spectrum at each point in time, or position along track. Since downlinking of raw radar data requires high bandwidth, an onboard processor that can generate the full Doppler spectrum in real time is desirable. This note describes design and testing of a prototype for such a processor, using field-programmable gate array (FPGA) technology (Ciletti 2003).

Corresponding author address: S. L. Durden, JPL MS 300-243, 4800 Oak Grove Dr., Pasadena, CA 91109.
E-mail: sdurden@jpl.nasa.gov

DOI: 10.1175/JTECH2086.1

2. Processor functional requirements

We design the Doppler processor so that it is able to process data collected by a future spaceborne radar system, such as that described by Fischman et al. (2005). We assume a pulse repetition frequency (PRF) of 6000 Hz and a system bandwidth of 4 MHz with 30-m range sampling. The CFT requires contiguous samples from radar footprints overlapping in the along-track direction, and so the Doppler data would likely be taken in a mode with fixed antenna pointing. A Doppler precision of better than 1 m s^{-1} is desired.

At each range bin the processor multiplies the complex radar data by a window function. Although a rectangular window function was used by Tanelli et al. (2004), better performance may be achievable with other window functions. Next, a fast Fourier transform (FFT) is calculated. This is followed by a magnitude-squared operation and averaging over eight range bins, providing an effective range resolution of approximately 240 m. The processor will handle 64 points along track for each spectrum. The 64 pulses are acquired in 11 ms at a PRF of 6000 Hz. Thus, real-time operation requires that all operations on 64 pulses be completed within 11 ms. Assuming sequential application of the processor to each range bin, the time available for processing is 22 μs per range bin. Here, 480 thirty-meter range bins have been assumed, giving a total range window of 14.4 km. The time for processing a given range

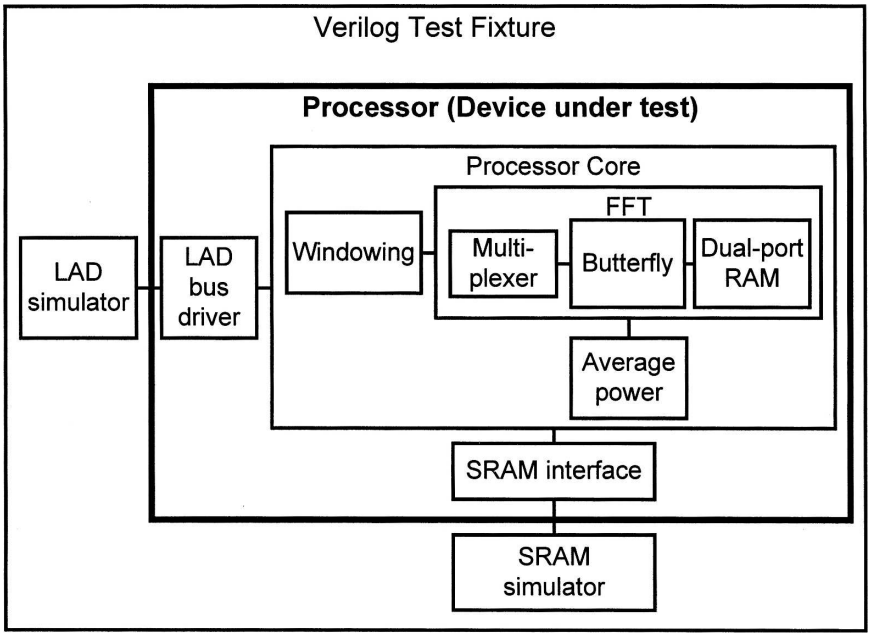

FIG. 1. High-level block diagram showing functional hierarchy of Verilog modules for both the processor and test fixture.

bin could be relaxed by processing two or more range bins in parallel, at the expense of more FPGA area.

One of the key requirements for the processor is dynamic range. Within a single range bin the output Doppler spectrum must accommodate a dynamic range of about 20 dB to allow an accurate velocity estimate. The processor is designed to accommodate an additional 40-dB dynamic range of total input power, based on a radar reflectivity from 15 to 55 dBZ. The overall dynamic range requirement for the processor output is thus approximately 60 dB. To achieve this dynamic range, theory indicates that the processor arithmetic must use at least 15 magnitude bits, in addition to a sign bit (Oppenheim and Weinstein 1972). This assumes an FFT implementation in which the butterfly outputs at each stage are right shifted by 1 bit (divided by 2) to avoid overflows at the next FFT stage (Ma 1997).

3. Processor implementation

The processor consists of the logic for performing the windowing, FFT, magnitude-squared, and eightfold averaging operations. To implement these processing functions in a way that can be used in both current and future technology, we use the Verilog Hardware Description Language (HDL), written at the register transfer level. Once implemented in Verilog, the design can be tested through simulation and can then be synthesized for a specific FPGA by running a synthesis software tool. The same Verilog code could also be used to synthesize an application-specific integrated circuit, although this is normally cost-effective only in high-volume applications. The design process using an HDL is analogous to the traditional method of developing a schematic. Like the schematic, the HDL description specifies the function of the circuit. Both the schematic and the HDL description can be implemented in a variety of technologies. An introduction to modern digital design using the Verilog HDL can be found in Ciletti (2003).

For laboratory testing of the Doppler processor, we used a custom FPGA board, previously used to demonstrate adaptive scan control and pulse-compression processing (Fischman et al. 2005). The block diagram in Fig. 1 shows the hierarchy of Verilog modules making up the processor and test fixture. The processor contains the core processing functions as well as modules for interfacing with the local address/data (LAD) bus and static random access memory (SRAM). Also shown in Fig. 1 is the Verilog test fixture for performing simulations prior to synthesizing the Verilog code for the FPGA. The test fixture instantiates the chip-level processor modules, as well as modules designed to simulate the LAD and SRAM. It also provides test input and diagnostic output for debugging. In laboratory testing using the FPGA board, only the "Processor" and its submodules are used to configure the

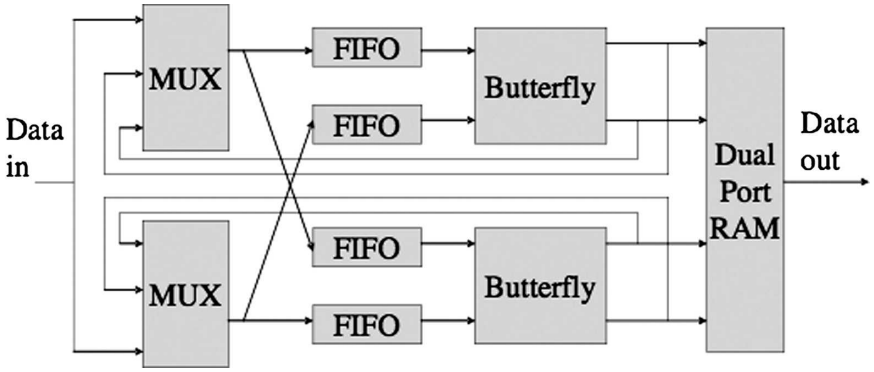

FIG. 2. Block diagram showing structure of the Verilog FFT module.

FPGA; the test fixture and LAD/SRAM simulation modules are not used.

Of the various modules in Fig. 1, the FFT module is the most complex. The FFT is a radix-2 decimation in frequency; it instantiates two butterflies, as shown in Fig. 2. The 64-point FFT is found by circulating the 64 samples through the two butterflies using first-in/first-out memory, based on a concept described by Singleton (1967). The parallelism generated by using two butterflies speeds up the FFT while still requiring a relatively small portion of the FPGA. The complete processor requires about 20% of the FPGA resources, for the case of the board and Xilinx Virtex-1000 FPGA described by Fischman et al. (2005).

4. Simulation and test results

Simulation of the Verilog processor description with the ModelSim logic simulation tool was used extensively for testing and debugging prior to laboratory testing. Tools such as ModelSim provide accurate simulation of digital circuits and are a major part of the design flow (Mentor Graphics 2004). Simulation provides the number of clock cycles required to process a single block of 64 pulses at all range bins. At the 20-MHz clock frequency used in laboratory testing, the time to process a single block of data (64 pulses, each with 480 range bins) is 6.5 ms, versus a requirement of 11 ms. For comparison, floating-point code to process a single

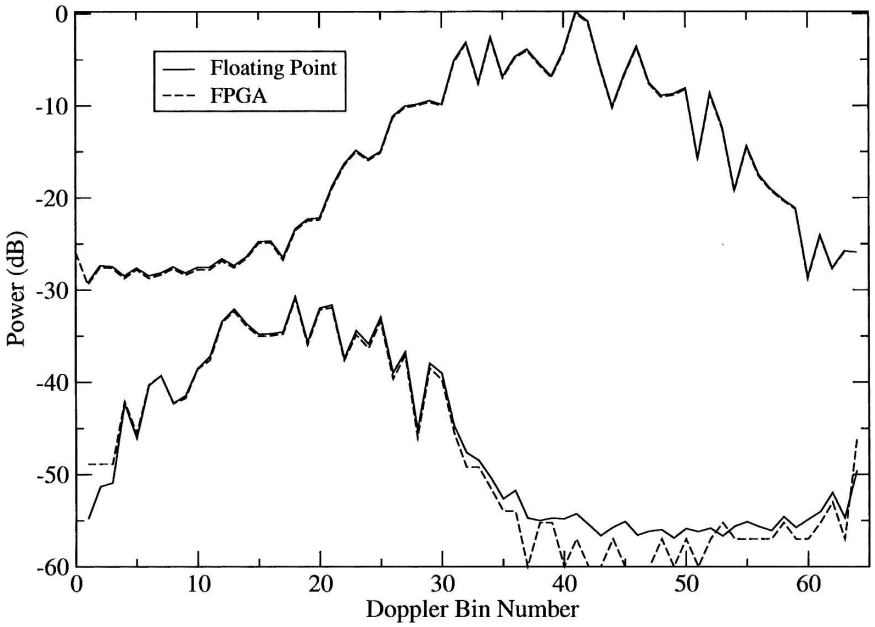

FIG. 3. Comparison of ARMAR data spectra from floating-point calculation and processing using the FPGA-based Doppler processor. Maximum unambiguous velocity is 25 m s^{-1}, corresponding to bin 33. Both strong and weak cases are shown.

block would require roughly 20 ms on a typical laptop computer. The output data rate of the processor is approximately 11 Mbits s^{-1}, versus 90 Mbits s^{-1} input rate.

Figure 3 shows the FPGA output along with a floating-point calculation for the same input data (one block of 64 pulses × 480 bins). The input data are raw data from the 14-GHz National Aeronautics and Space Administration/Jet Propulsion Laboratory (JPL) Airborne Rain Mapping Radar (ARMAR). These data were acquired with bandwidth and PRF that are close to those assumed in section 2; ARMAR is described in detail in Durden et al. (1994). The maximum unambiguous Doppler velocity in Fig. 3 is 25 m s^{-1} and corresponds to bin 33. The Doppler velocity spectra are broad because of aircraft motion; some signal is present in all Doppler bins for the stronger signal. The FPGA and floating-point spectra in Fig. 3 are indistinguishable for the stronger signal. The comparison is also good for the weaker signal, although some differences can be seen.

To test the FPGA-based processor further, we created a much larger set of input data consisting of 230 blocks. The dataset was generated using the technique of Zrnic (1975) to simulate realistic raw data from Doppler spectra. The Doppler spectra were generated for each range bin in each 64-pulse block by simulating a spaceborne Doppler radar overflying a mesoscale convective system generated by a cloud-resolving model (Tanelli et al. 2002, 2004). Table 1 provides statistics of the input data reflectivity Z and the first and second Doppler velocity spectral moments μ_v and σ_v. The theoretical σ_v for a uniformly filled beam is 11 m s^{-1}; the extremes in Table 1 are generated by nonuniformity within a beam.

Figure 4 shows the results of processing the raw radar data in both floating point and using the fixed-point FPGA processor. Output spectra from both types of processing were used as input to the floating-point CFT algorithm to estimate the Doppler mean. Scatterplots labeled "Spectrum" are derived by computing the normalized bias and RMS difference over the full spectrum for each range bin and block. These are defined by

TABLE 1. Statistics of reflectivity, spectral mean, and spectral width of 230-block input dataset.

	Z (dBZ)	μ_v (m s^{-1})	σ_v (m s^{-1})
Mean	27.68	1.98	11.34
Std dev	11.24	6.63	2.72
Min	1.35	−19.23	3.84
Max	51.52	27.18	25.39

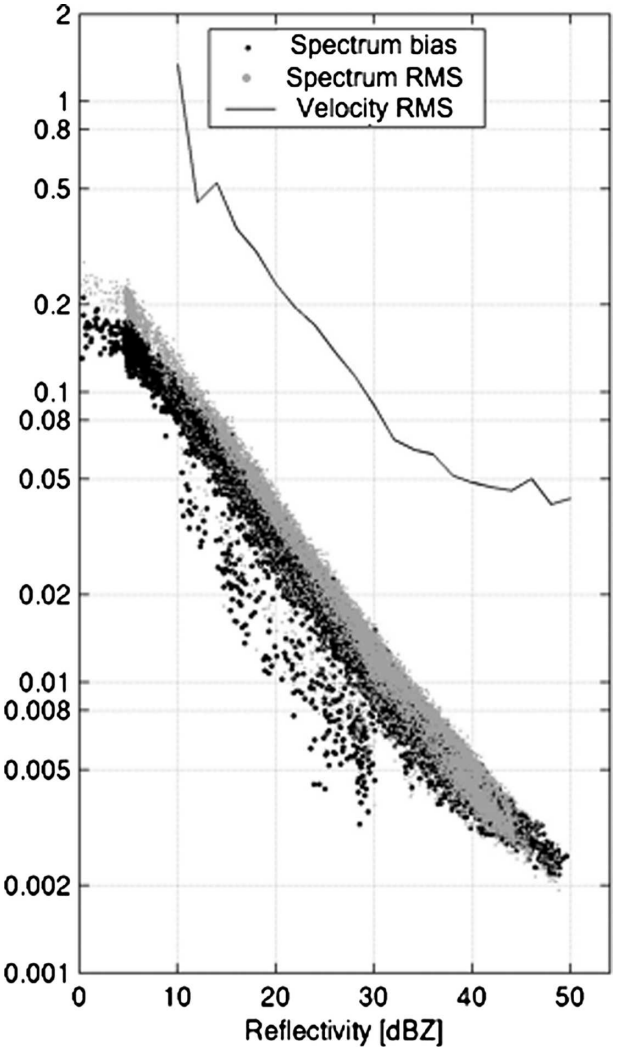

FIG. 4. Comparison of FPGA and floating-point processing as a function of input reflectivity using spaceborne radar data simulated from model output. Vertical scale is normalized difference for spectra (unitless) and velocity difference (m s^{-1}) for velocity plot.

$$\text{bias} = \frac{\sum_i (P_i - \overline{P}_i)}{\sum_i P_i} \quad \text{and}$$

$$\text{RMS} = \left[\frac{\sum_i (P_i - \overline{P}_i)^2}{\sum_i (P_i)^2} \right]^{1/2},$$

where P_i is the measured power in Doppler bin i, and $\overline{P}_i$ is the corresponding infinite precision power. The bias and RMS difference are unitless.

The velocity plot is the RMS difference between the floating-point and FPGA Doppler velocity estimates,

with units of meters per second. The RMS difference is 0.5 m s^{-1} or less for input reflectivities from 14 to 50 dBZ, indicating an input dynamic range of over 35 dB. Because of a workmanship defect in the test board's most significant bit, input was limited to 14 magnitude bits (and one sign bit). Since the processor was designed to handle a maximum input with 15 magnitude bits, the defect resulted in a reduction in the maximum input signal by 6 dB. Hence, the processor has an additional 6 dB of dynamic range not used in these tests. The total input dynamic range is roughly 40 dB, as desired, and the total dynamic range is roughly 60 dB. This is also supported by the results in Fig. 3.

5. Conclusions

We have described an FPGA implementation of a Doppler spectral processor that could be used in a spaceborne precipitation radar application. The FPGA approach takes advantage of the capabilities (speed and density) of today's FPGAs. The design using the Verilog HDL, rather than more traditional schematics, allows easy simulation and modification of the processor hardware. Simulation and laboratory testing indicate that the processor meets performance goals.

Acknowledgments. The authors thank A. Berkun of JPL for providing his expertise in system testing and debugging. The research described here was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the U.S. government or the Jet Propulsion Laboratory, California Institute of Technology.

REFERENCES

Amayenc, P., 1993: Proposal for a spaceborne dual-beam rain radar with Doppler capability. *J. Atmos. Oceanic Technol.,* **10,** 262–276.

Ciletti, M. D., 2003: *Advanced Digital Design with the Verilog HDL*. Prentice-Hall, 982 pp.

Doviak, R. J., and D. S. Zrnic, 1993: *Doppler Radar and Weather Observations.* 2d ed. Academic Press, 562 pp.

Durden, S. L., E. Im, F. K. Li, W. Ricketts, A. Tanner, and W. Wilson, 1994: ARMAR: An airborne rain-mapping radar. *J. Atmos. Oceanic Technol.,* **11,** 727–737.

Fischman, M. A., A. C. Berkun, W. Chun, E. Im, and R. Andraka, 2005: An onboard processor and adaptive scanning controller for the second generation precipitation radar. *IEEE Trans. Geosci. Remote Sens.,* **43,** 802–812.

Ma, Y., 1997: An accurate error analysis model for fast Fourier transform. *IEEE Trans. Signal Process.,* **45,** 1641–1645.

Meneghini, R., and T. Kozu, 1990: *Spaceborne Weather Radar.* Artech House, 199 pp.

Mentor Graphics, 2004: ModelSim SE: Advanced simulation and debug datasheet. 3 pp. [Available online at http://www.mentor.com/products/fv/digital_verification/modelsim_se/upload/se.pdf.]

Oppenheim, A. V., and C. J. Weinstein, 1972: Effects of finite register length in digital filtering and the fast Fourier transform. *Proc. IEEE,* **60,** 957–976.

Singleton, R. C., 1967: A method for computing the fast Fourier transform with auxiliary memory and limited high-speed storage. *IEEE Trans. Audio Electroacoust.,* **15,** 91–98.

Tanelli, S., E. Im, S. L. Durden, L. Facheris, and D. Guili, 2002: The effects of nonuniform beam filling on vertical rainfall velocity measurements with spaceborne Doppler radar. *J. Atmos. Oceanic Technol.,* **19,** 1019–1034.

——, ——, ——, ——, ——, and E. A. Smith, 2004: Rainfall Doppler velocity measurements from spaceborne radar: Overcoming NUBF effects. *J. Atmos. Oceanic Technol.,* **21,** 27–44.

Zrnic, D. S., 1975: Simulation of weatherlike Doppler spectra and signals. *J. Appl. Meteor.,* **14,** 619–620.