A Case Implementation of a Spotlight Range Migration Algorithm on FPGA Platform

A. Melnikov[1], J. Le Kernec[2], D. Gray[3]
[1]EIE Department, Hong Kong Polytechnic University, Hong Kong
[2]EEE Department, University of Nottingham, Ningbo, China
[3]EEE Department, Xi'an Jiatong Liverpool University, Suzhou, China

Abstract – This paper presents a real-time implementation of a spotlight range migration algorithm processor on FPGA platform for the MIT open courseware radar. The modified radar platform is presented. The paper also describes the use of FPGA resources on a Virtex-6 DSP kit board, and compares the results obtained with the hardware implementation and the Matlab equivalent. A good match between the real-time and the offline processing of data was found.

***Index Terms* — SAR, RMA, FPGA, DSP, radar.**

I. INTRODUCTION

This project is based on the MIT open course project "Build a small radar system capable of sensing Range, Doppler and Synthetic Aperture Radar Imaging" [1].

This laptop-based radar is capable of performing SAR Imaging. FPGA platforms like Virtex-6 DSP kit [2] could be used to replace the laptop and provide onboard real-time processing capability, for example, on a multi-rotor UAV in flight to reduce the data sent to the control station for applications such as urban mapping and machine vision.

In Section II, the modified MIT radar frontend is presented. In Section III, the spotlight range migration algorithm (SRMA) is presented. Section IV discusses the implementation of the SRMA on FPGA. Finally, Section V compares the SAR images obtained using MATLAB against the real-time implementation on a Virtex-6 FPGA board.

II. RADAR FRONTEND

The radar front end used was derived from the MIT radar [1]. The modifications to the circuit were presented in [3]. The characteristics of the radar system are shown in Table I. The experimental frontend is shown in Fig.1.

TABLE I
RADAR CHARACTERISTICS

Waveform	Up- and down-chirp (triangular)
Waveform Synthesis	Triangular wave to VCO
Bandwidth/ Radio frequencies	92MHz/ [2.405-2.497MHz]
Operation/ Pulse repetition period	FMCW/ 40ms
Mode/Antenna	Bistatic/ Endfire dipole cantenna
Gain/Polarization	8dBi/VV
Sampling Frequency	44.1kHz

III. SPOTLIGHT RANGE MIGRATION ALGORITHM

Along its track, the radar transmits and receives a train of pulses while moving. For each pulse transmitted, the radar assumes a new along-track position. The numbers of elements that constitute the "synthetic array" are a function of the antenna beam pattern and the distance between measurements.

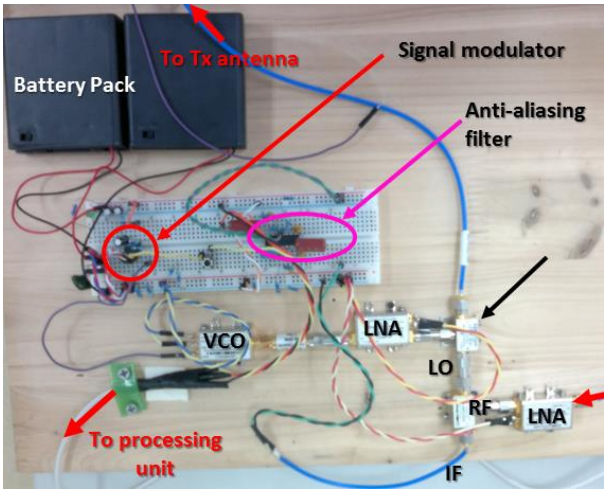

Fig. 1. SAR imaging system.

A narrow azimuth beamwidth, for cross-range is produced by coherently processing the radar backscattered signals. This adds another dimension and allows the generation of 2D images (range and cross-range).

Spotlight Range Migration Algorithm (SRMA) was used for SAR imaging, which compared to some other SAR algorithms (Polar Formation Algorithm, for instance) has an important advantage in that the illuminating wavefronts are not assumed to be planar, which enables avoidance of space-variant defocusing and geometric distortion of the final image. The image processing can be split in two parts: pre-processing and image formation.

A. Pre-processing

Before the image formation matrix is created, where the rows will be populated with the echo signals received by different synthetic array elements and the columns will contain those scatterings in the range direction. These are then processed using a Hilbert transform and the complex output is fed to the image processor.

B. Image formation

Image Formation Processor (IFP) for the SRMA comprises 4 stages: along-track FFT, Matched filtering, Stolt interpolation and 2D iFFT, as detailed in [4]. The results of the modified algorithm compared to the original were shown in [3] and were found to be equivalent.

IV. FPGA IMPLEMENTATION OF SRMA

The raw data was fed to the Pre-Processor (PP) where the up-chirps were extracted. 8 chirps were averaged per array element. Then the data went through a clutter canceller and before being zero-added for the FIR implementation of the Hilbert transform [4].

The output of the Hilbert transform went to the IFP. The signal is first windowed (e.g. Hanning). The matrix was then transposed and zero-padded for along-track FFT. The result was matched filtered against coefficients stored in memory. The next step is the Stolt interpolation [4] which corrected the range curvatures for scatterers at different ranges. After interpolation, windowing was again applied before the 2D-iFFT to finally generate the image.

Note that the second part of the SAR processor performed operations in both range and cross-range dimensions, and the data flow rearrangement required extensive usage of the memory. This imposed limitations on the maximum amount of data being processed, since the depth of the RAM blocks in the Virtex-6 board were limited to 64 kB [5].

Since every array element contained 882 samples and there were 55 array elements overall, the maximum possible FFT size in the cross range direction was 64, which was held in the available memory to rearrange the signal order from range to cross-range wise flow (882×64= 56 kB < 64 kB), but a small FFT length in the cross-range restricted the resolution of the final image (see Fig.2).

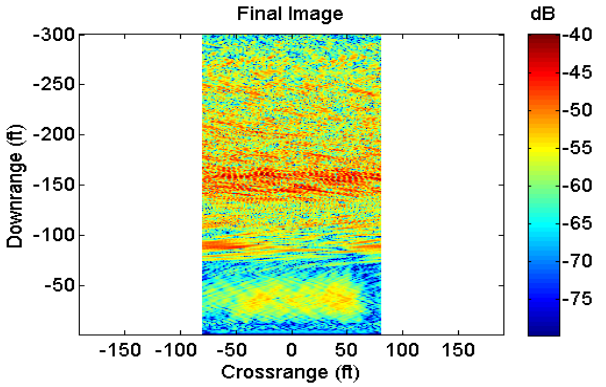

Fig. 2. HW SAR processor result for "Back of Warehouse" data from [1].

Table II summarizes the characteristics of the processors mapped on the target platform. From the results in Table II, the FPGA platform Virtex-6 was capable of processing the data in real-time for SAR pre-processor. The image processor needed to be redesigned to fit on Virtex-6.

V. COMPARING THE RESULTS FROM MATLAB AND FPGA PROCESSOR

Both software (SW) and hardware (HW) implementations use the original datasets provided in [1] for the validation of the HW implementation. The results for the PP and the IFP are discussed below.

Fig. 3 shows that the complex output relative error to software processing for the preprocessor.

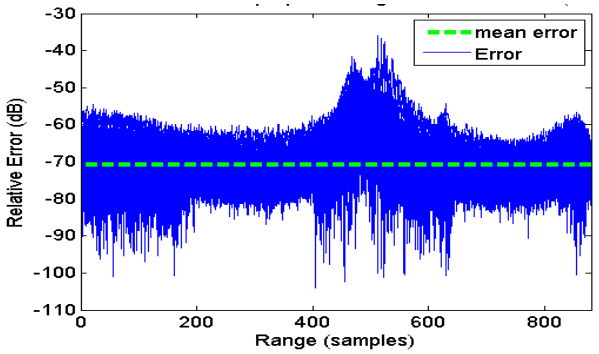

Fig. 3. Relative error after the Preprocessor Stage (before the Image Formation Stage).

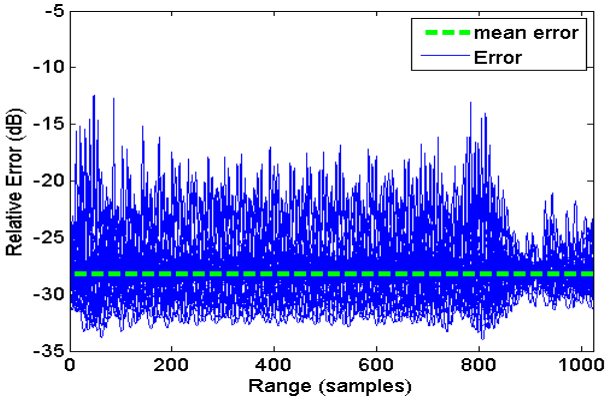

Fig. 4. Relative error between Matlab and FPGA SAR images.

TABLE II
FPGA HARDWARE IMPLEMENTATION SYNTHESIS

Processor	F_{max} (MHz)	Slice registers	Slice LUTs	Memory	IO
PP	86	1%	9%	9%	57%
IFP	n/a	5%	10%	138%	114%

VI. CONCLUSION

This paper described the implementation of the radar front end and its real-time SRMA processor on an FPGA platform. The preprocessor on FPGA yielded a mean error compared to its SW equivalent of 70.7dB and -36dB at most. The image formation processor had a mean error compared to its SW equivalent of 28.2dB and -12.5dB at most. Further work will be required to fit the IFP on the chip and reduce the relative mean error to at least -40dB. Some thoughts on getting closer to integration: mapping the Hilbert Transform and FFT functions on the DSP slices while making use of Multiply Adder IP from Xilinx might significantly reduce the use of IOs and exploiting symmetry for the FIR and FFT could significantly reduce memory usage.

REFERENCES

[1] G. Charvat, et al., "RES.LL-003 Build a Small Radar System Capable of Sensing Range, Doppler, and Synthetic Aperture Radar Imaging", *MIT OpenCourseWare: Massachusetts Institute of Technology*, January 2011.

[2] www.xilinx.com, Spartan-3A starter kit & Virtex-6 DSP kit.

[3] A. Melnikov, J. Le Kernec and D. Gray, "FMCW Rail-mounted SAR: porting spotlight SAR imaging from MATLAB to FPGA," *IEEE Int. Conf. on Signal Process., Commun. & Comput.* (ICSPCC 2014), Guilin, August 2014, paper #2119.

[4] W. Carrara, R. Goodman and R. Majewski, *Spotlight Synthetic Aperture Radar Signal Processing Algorithms*, Artech House, 1995.

[5] Xilinx, "UG363. Virtex-6 FPGA Memory Recources", 2014, www.xilinx.com