Design of FPGA on-chip module for real-time image processing

Limin Li[1] and Alice M. Wyrwicz[1,2]

[1]Center for Basic MR Research, NorthShore University HealthSystem, Evanston, IL, United States, [2]Department of Biomedical Engineering, Northwestern University, Evanston, IL, United States

Target Audience: MR physicists and engineers interested in real-time image processing.

Purpose: MR imaging speed is determined by the rates of data acquisition and image processing. MRI systems equipped with a multi-channel data-acquisition module[1] can collect a huge amount of data within a short time. In order to obtain the full benefit of these data-acquisition capabilities, it is desirable to have more sophisticated hardware capable of running increasingly complicated image processing algorithms at comparable speeds or in real time. In this work, we describe the design and testing of a real-time image processing module on a single-chip Field-Programmable Gate Array (FPGA).

Design: The image processing module was built on a Xilinx FPGA chip XC6SLX45. The FPGA chip sits on a NI sbRIO-9636 board (National Instruments, Inc., Austin, TX USA). FPGAs are usually designed by using standardized Hardware Description Languages (HDLs) such as VHDL and Verilog. However, directly writing HDL programs is time consuming and requires considerable experience. Our design was implemented in LabView platform (2013 version). The module consists of data pre-processing and image reconstruction. The pre-processing consists of baseline correction, lowpass filtration and decimation. The image reconstruction is carried out by a 2D FFT core. Our design of the 2D-FFT core is distinguished from the previously reported work[2-4] in two respects. No off-chip hardware resources are required, which increases portability of the core. Matrix transposition usually required for execution of 2D FFT is completely avoided by using a newly-designed address generation unit (AGU). This saves a large amount of on-chip memory and many clock cycles. The 2D FFT is executed in two loops, First FFT Loop and Second FFT Loop (Fig. 1). All processing operations within a loop are completed in a single clock cycle (16.67ns). A FIFO (first-in, first-out) buffer ToFPGA allows input data for entering First FFT Loop; while the processed data from Second FFT Loop are transferred to a host PC via another FIFO buffer ToHost.

Fig. 1: LabView code for execution of 2D FFT.

Results and Discussion: We tested the image processing module by reconstructing multi-slice images. A graphical user interface (GUI) program was developed using the LabView platform to initialize and start the FPGA board and manage data transfer between the FPGA and the PC, which communicate via a local network. In the tests on real-time streaming data, analog signals were generated by sending raw data to the DAC chips on the FPGA board. By hardwiring the analog outputs to the analog inputs, the streaming data were produced when the analog signals were sampled with 16bit ADCs at a rate of 100kS/s. In the tests on static data, the raw data were transferred directly to the 2D FFT core. Typical images reconstructed with the module from a single slice of an 8-slice dataset from a rabbit brain are shown in Fig. 2 (left, middle). Note that the images are comparable to conventional software reconstruction (Fig. 2, right) with the PC. For a 128×128 image, the processing rate is 2.5ms, equivalent to 400 frames/sec.

Fig. 2: Animal brain images reconstructed from simulated real-time echo signals (left) and static data (middle) using the FPGA, and the same raw data using a PC (right).

Summary: The results demonstrate that the FPGA on-chip image processing module works as expected. Future work will integrate the image processing module with a digital receiver on an FPGA.

References: 1. Schmitt M, et. al. MRM 2008; 59: 1431-1439. 2. Dalal IL, et. al. presented at the 40th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA. (2007). 3. Tang W and Wang W. Meas. Sci. Technol. 2011; 22: 015902. 4. Vistnes KE, et al. presented at the 19th International Parallel and Distributed Proce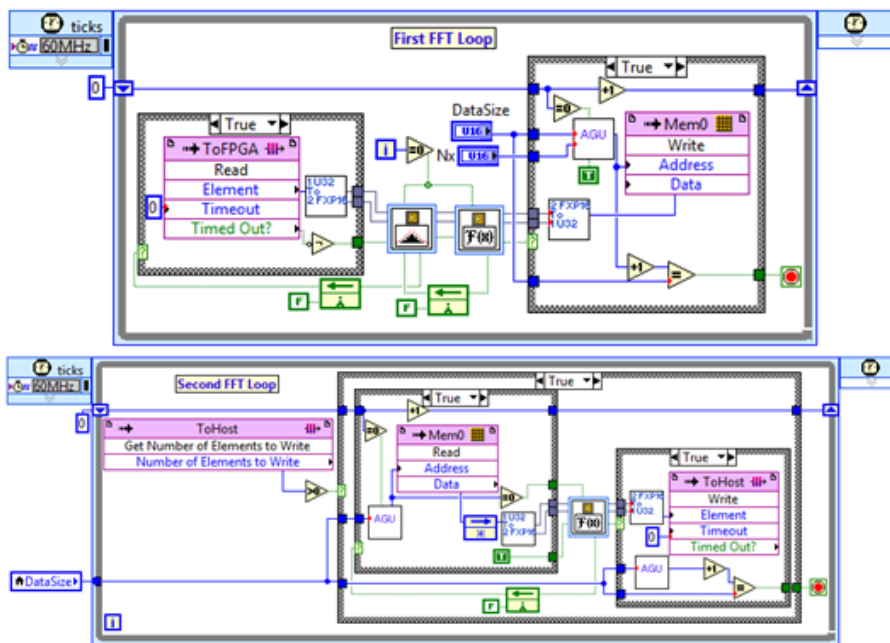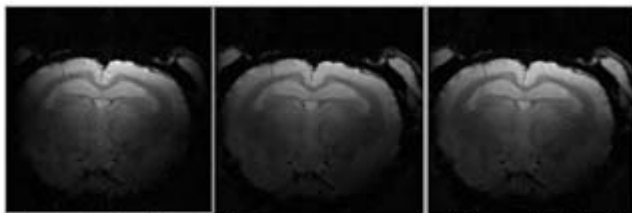ssing Symposium, 4-8 April 2005, Denver, CO, USA.