Optimization of real-time ultrasound PCIe data streaming and OpenCL processing for SAFT imaging

M. Walczak, M. Lewandowski, N. Żołek
Department of Ultrasound
Institute of Fundamental Technological Research PAS
Warsaw, Poland
mlew@ippt.pan.pl

***Abstract*—Our goal is to develop a complete ultrasound platform based on real-time SAFT (Synthetic Aperture Focusing Technique) GPU processing. We are planning to integrate all the ultrasound modules and processing resources (GPU) in a single rack enclosure with the PCIe switch fabric backplane. The first developed module (RX64) provides acquisition and streaming of 64 ultrasound channels. We implemented and benchmarked data streaming from the RX64 to the GPU memory and the SAFT image reconstruction on the GPU. A high system performance was achieved using hardware assisted direct memory transfers and pipelined processing workflow. The complete system throughput, including 128 channel data transfer at 16kS per line and low-resolution 256x256 pixel image SAFT reconstruction on a single Nvidia K5000 GPU, reached 450 fps. The obtained results proved the feasibility of the ultrasound real-time imaging system with GPU SAFT processing.**

Keywords— ultrasonic imaging; synthetic aperture; GPGPU; FPGA

I. INTRODUCTION

Medical ultrasound based visualization systems require a vast amount of data bandwidth and computational power for real-time execution of the signal analysis and imaging algorithms. Thus so far the front-end processing is realized using hardware solutions. However, hardware based processing become a barrier for implementation of more and more complex imaging algorithms e.g. based on synthetic aperture imaging. In recent years, a dynamic development of parallel processing technology, especially multi-core processors (CPU) and general purpose graphics processing units (GPGPU) enabled migration of hardware based signal processing to more flexible software based signal processing. This trend is clearly visible in literature, where numerous studies are devoted to GPGPU applications in ultrasound signal processing [1-4]. However, acquisition and communication architecture of the system are equally important.

We took a holistic approach to the system architecture in order to smoothly integrate a hardware acquisition subsystem and asynchronous software based processing.

A real-time implementation of the Synthetic Aperture Focusing Technique (SAFT) imaging methods is a big engineering challenge due to the required extremely high data bandwidth and performance of data processing [5-8]. Nowadays, only a complex field programmable gate arrays (FPGA) and GPGPUs offer a processing power sufficient to implement such tasks. Still, care must be taken to ensure balanced resource utilization and communication bandwidth.

A versatile ultrasound acquisition and processing platform designed in our lab will enable implementation of the SAFT methods and other complex algorithms of ultrasound signal processing and visualization. The developed RX64 card provides the acquisition as well as streaming of 64 parallel ultrasound channels through the 2nd generation 8-lane PCIe interface. The objective of this work was to determine maximum sustained raw data throughput from the RX64 via CPU memory to the GPU memory, as well as the performance of the GPU implementation of the SAFT reconstruction algorithm on real-time streamed data.

II. SYSTEM DESIGN

A. System Architecture

The system architecture (Fig. 1) is based on the standard PCIe switched fabric. The designed and built a 64-channel acquisition module (RX64) is equipped with 2nd generation 8 lane PCIe communication interface. The RX64 contains a high-end FPGA Stratix IV 70 GX (Altera, USA) interfaced to: two 32-channels mixed-signal front-end modules SMM913x (Cephasonics, USA) and two 64-bit 8 GB DDR3 SO-DIMM memories for data buffering. The Cephasonics modules provide analog conditioning of the ultrasound echoes, A/D conversion with 12-bit resolution at 65 MSPS, and data serialization.

Serialized 64-channels data are transferred to the FPGA and after deserialization are stored in the local DDR3 memory. Then, the data are transferred through the PCIe interface to the CPU RAM memory. The internal (64-ch ADC: 50 Gbps, DDR3: 128 Gbps) and external (PCIe: 40 Gbps) interface

bandwidth of the RX64 were balanced to meet real-time streaming requirements.

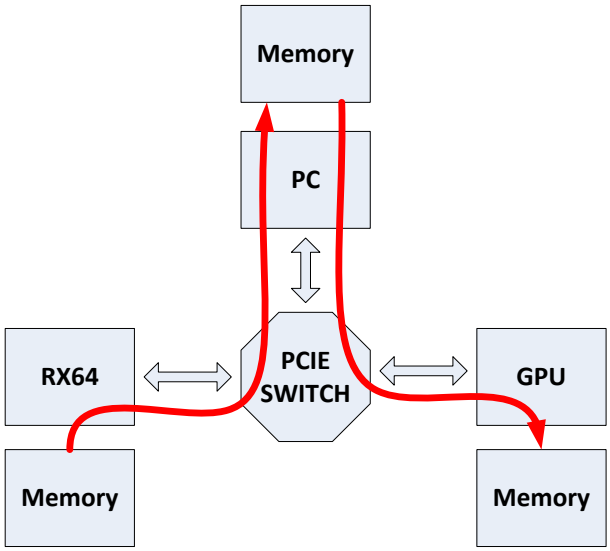

Figure 1. The block diagram of RX64->GPU data streaming.

Data streaming from the module is performed by custom made DMA (Direct Memory Access) engine which provides very high utilization of the PCIe throughput. The DMA transfers data packets between DDR3, that is local to the FPGA and PC memory, without involving CPU. The data transfers are initiated from the CPU by setting up a number of transfer descriptor records that are maintained in the FPGA internal memory. Afterwards the engine simultaneously reads data from two DDR3 memories and assembles PCI Express posted write packets that transfer data to CPU memory. After the transfer is finished an interrupt is generated to signal the CPU. The DMA engine, acting as a PCIe bus-master, delivers maximum performance with minimal CPU load. Data are transferred to a page-locked CPU memory, i.e. mapped to a fixed physical address.

B. GPU based processing

The computing of the acquired data is performed using GPGPU processor and OpenCL framework. The data transfer from the CPU to the GPU memory with optimal use of the OpenCL functions eliminates additional copying of data within the CPU memory. Therefore a direct use of a memory region, which was allocated by the transfer from the RX64, is possible ("pinned memory"). In order to achieve the highest throughput, all transfers are implemented as asynchronous operations. The transfers from RX64 to the CPU memory, from CPU memory to GPU memory as well as SAFT kernel [9] execution operate simultaneously (pipelining processing). Output from processing kernel is displayed as a texture with use of OpenGL.

Processing and copying the input and output data use two OpenCL queues and allow to minimize time lags between consecutive computational kernels execution and optimally use of GPGPUs' compute capability version 2.0 or higher [10, 11]. The output data from previous kernel execution are copied back to the CPU memory simultaneously with writing to GPU the input data for the next kernel processing. Simultaneously processing of the data previously transferred is executed. This asynchronous process is shown schematically in Fig. 2. The time delays between kernel executions occur when drawing is executed with use of the same graphics processor due to synchronization with OpenGL buffers even when using OpenCL – OpenGL interoperability feature.

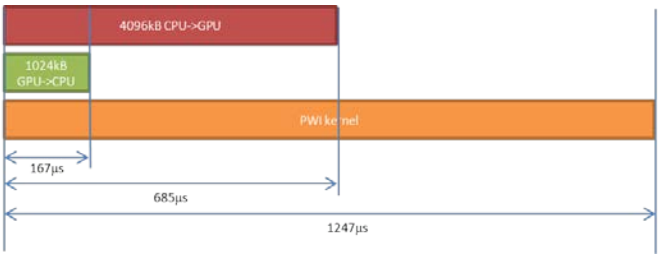

Figure 2. Scheme of the part of OpenCL execution timeline with asynchronous data transfer (to and from GPU memory) and Synthetic Aperture kernel execution.

The kernel processing durations depend on applied reconstruction algorithm. The presented exemplary computing kernel is based on Synthetic Aperture method which utilizes general plane wave (PW) data [8]. The data for reconstruction are collected from 128 channels after insonification, where all 128 elements generate plane wave propagating inside the medium.

A Single PW image is obtained with use of additional apodization [12] on the detection side. The apodization function was approximated with polynomial approximation to minimize calculation of trigonometric functions on GPGPU. Such approximation gives almost two times shorter kernel execution time in comparison to kernel with original apodization function. For additional optimization of the kernel, the suggestions from [13] and strength reduction [14] were applied. All optimizations preserve the same level of accuracy as the algorithm without optimizations.

III. RESULTS

All tests were performed on the PC-class computer with the Intel i7-960 (3.2 GHz) processor equipped with 6 GB of RAM, ASUS-P5T7WS motherboard and the GPU NVIDIA Quadro K5000 and running Microsoft Windows 7 64-bit operating system.

The applied asynchronous transfer of data between the RX64 CPU and RAM has reached 3 GB/s (theoretical limit is 3.8 GB/s). Transfer time of a single frame of the SAFT acquisition using 128 channels, sampling at 65MSPS, imaging depth of 19 cm (128ch x 16kS x 2B = 4 MB) was approximately 1.3 ms. That provides the acquisition frame rate of up to 769 Hz to the CPU memory. The transfer of data from the CPU memory to the GPU memory was 5.7 GB/s. The whole RX64-CPU-GPU transfer reached 3 GB/s and is limited mainly by the RX64-CPU. We observed that in terms of transfer throughput in this particular configuration (1x RX64, 1x GPU) the RX64-CPU-GPU transfer is as good as direct RX64-GPU transfer.

The obtained real-time data transfer from the CPU to the GPU and reconstruction of the Low Resolution Image (LRI) at resolution of 256x256 pixels, on a single GPU card allow to obtain frame rate at level approximately 450 Hz.

TABLE I. THE OBTAINED DATA THROUGHPUT

Parameter Name	***Frame rate [fps]***
GPU SAFT image proccesing only (LRI 256x256 pixels)	800
RX64->CPU data transfer only (128-channlels data @ 16kS/line)	769
CPU->GPU data transfer only (128-channlels data @ 16kS/line)	1459
RX64->CPU->GPU data transfer and GPU SAFT image proccesing (LRI 256x256 pixels)	450

The exemplary visualization of the analyzed data is shown in Fig. 3.

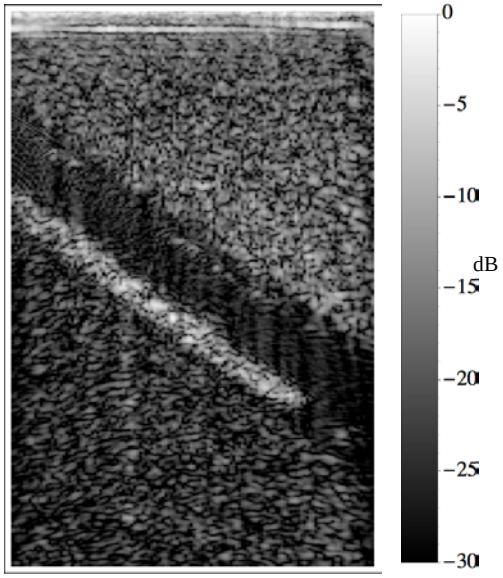

Figure 3. Example of reconstructed image using plane wave imaging with data acquired from a phantom; insonification angle was equal to 0 degrees and apodization was applied on the detection side. Visualization depth is equal to 7 cm.

Although 450 fps is reasonable processing speed for applications, the single SAFT kernel execution time is approximately 1.5 ms and with display synchronization delays (even when using OpenCL-OpenGL interoperability [10]) is the major frame-per-second bottleneck.

IV. CONCLUSIONS

The new ultrasound system architecture and the processing flow enable multi-channel real-time ultrasound acquisition, streaming and GPU processing. The described implementation of the ultrasound PCIe data streaming and GPU processing proves the feasibility of ultrasound real-time imaging with the GPU SAFT processing for systems with 64–128 channels. The presented system uses commercial-of-the-shelf components as a computing platform, thus lowers a total system cost. New ultrasound processing algorithms can be easily implemented on the GPU using widely available development tools.

The described RX64 acquisition module is the first element of the ultrasound versatile platform being in a process of development. In the next step, the transmit module will be developed and integrated into the system. The GPU software framework, based on pipelined streaming and processing will be extended with subsequent kernels for the ultrasound processing (including flow velocity imaging [15] etc.) as well as optimized for new version 5 of the Nvidia CUDA.

We have hopes to integrate the complete 192-channel ultrasound system in one year. The system has a potential to become a tool for both educational and research purposes and introduce new applications and developments in the ultrasound field.

ACKNOWLEDGMENT

Project POIG.01.03.01-14-012/08-00 co-financed by the European Regional Development Fund under the Innovative Economy Operational Programme.

REFERENCES

[1] M. Lewandowski, "Medical Ultrasound Digital Signal Processing in the GPU Computing Era", in *Computer Vision in Medical Imaging*, ed.C.H. Chen, Word Scientific, 2013 (in press).

[2] J. Chen, B. Y. S. Yiu, and A. C. H. Yu, "Medical Ultrasound Imaging : To GPU Or Not To GPU ?," Micro, IEEE , vol.31, no.5, pp.54-65, Sept.-Oct. 2011.

[3] S. Kim, H. Sohn, J. H. Chang, T. Song, and Y. Yoo, "A PC-based fully-programmable medical ultrasound imaging system using a graphics processing unit," *2010 IEEE International Ultrasonics Symposium*, pp. 314–317, Oct. 2010.

[4] M. di Bisceglie, M. di Santo, C. Galdi, R. Lanari, and N. Ranaldo, "Synthetic Aperture Radar Processing with GPGPU," *IEEE signal processing*, vol. 27, no. 2, 2010.

[5] L. Grønvold, Implementing Ultrasound Beamforming on the GPU using CUDA, Master Thesis, Norwegian University of Science and Technology, Department of Engineering Cybernetics, 2008.

[6] B.Y.S. Yiu, I.K.H. Tsang, A.C.H. Yu, Real-time GPU-based software beamformer designed for advanced imaging methods research, Ultrasonics Symposium (IUS), 2010 IEEE, pp.1920–1923, 11–14 Oct. 2010.

[7] M. Lewandowski, M. Walczak, B. Witek, P. Kulesza, K. Sielewicz, "Modular & Scalable Ultrasound Platform with GPU Processing", Ultrasonics Symposium (IUS), 2012 IEEE, 7–10 Oct. 2012.

[8] B.Y.S. Yiu, I.K.H. Tsang; A.C.H. Yu, GPU-based beamformer: Fast realization of plane wave compounding and synthetic aperture imaging, Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on, vol.58, no.8, pp.1698–1705, 2011.

[9] M. Lewandowski, P. Karwat, J. Kudelka, and T. Kleczek, "GPU Implementation of the STA Algorithm on I/Q Data," Ultrasonics Symposium (IUS), 2012 IEEE, 7–10 Oct. 2012.

[10] B. R. Gaster, L. Howes, D. Kaeli, P. Mistry, and D. Schaa, *Heterogeneous Computing with OpenCL*. 2012.

[11] NVIDIA, OpenCL Best Practices Guide. 2011.

[12] Y. Tasinkevych, I. Trots, A. Nowicki, and P. Lewin, "Modified synthetic transmit aperture algorithm for ultrasound imaging.," *Ultrasonics*, vol. 52, no. 2, pp. 333–42, Feb. 2012.

[13] Intel, "Writing Optimal OpenCL ™ Code with Intel ® OpenCL SDK," pp. 1–38.

[14] J. Cocke and K. Kennedy, "An Algorithm for Reduction of Operator Strength," Communications of the ACM, vol. 20, no. 11, pp. 850–856, 1977.

[15] L.W. Chang, K.H. Hsu, P.Ch. Li, Graphics processing unit-based high-frame-rate color doppler ultrasound processing, IEEE TUFFC, vol.56, no.9, pp.1856–1860, 2009.