Title	Design of a programmable micro-ultrasound research platform
Author(s)	Chiu, HCT; Zhang, L; Cheung, DKH; Hu, C; Shung, KK; Yu, ACH
Citation	The 2010 IEEE International Ultrasonics Symposium, San Diego, CA., 11-14 October 2010. In Proceedings of IEEE IUS, 2010, p. 1980-1983
Issued Date	2010
URL	http://hdl.handle.net/10722/129679
Rights	Proceedings of the 2010 IEEE International Ultrasonics Symposium. Copyright © IEEE.

10.1109/ULTSYM.2010.0500

Design of a Programmable Micro-Ultrasound Research Platform

Harry C. T. Chiu*, Lequan Zhang†, Dave K. H. Cheung*, Changhong Hu†,
K. Kirk Shung†, and Alfred C. H. Yu*

* Medical Engineering Program, The University of Hong Kong, Pokfulam, Hong Kong SAR
† Ultrasonic Transducer Resource Center, The University of Southern California, USA

Corresponding Email: alfred.yu@hku.hk

***Abstract*—To foster innovative uses of micro-ultrasound in biomedicine, it is beneficial to develop flexible research-purpose systems that allow researchers to easily reconfigure its system-level operations such as transmit firing sequence and receive processing. In this paper, we present the development of a programmable micro-ultrasound research platform that is capable of realizing various micro-imaging algorithms. The research platform comprises a linear-array-based scanning front-end and a PC-based data processing back-end, which employs a graphical processing unit (GPU) as the processor core. The front-end operations can be configured from the PC via the parallel port and the two blocks are synchronized by an external clock. Acquired data from the front-end is first digitized and relayed to the PC through an data acquisition card (200 MHz, 14-bit). They are then transferred to the GPU (GTX 275) in which the image formation is carried out via multi-thread processing. Results are displayed on-screen in real-time and can be saved to the PC's hard disk for offline analysis. Through a module-based programming approach, this platform can facilitate realization of custom-designed imaging algorithms developed by researchers. In this work, B-mode imaging and adaptive color flow imaging have been implemented as demonstrations of the research platform's programmability. The performance results show that real-time processing frame rates can be achieved for both imaging modes.**

Keywords—micro-ultrasound, research platform, programmability, array system.

I. Introduction

Non-invasive visualization of living tissues with micro-scopic resolution has long been attracting scientific and clinical attention in studying both healthy and disease processes. For instance, monitoring of small animal models could provide valuable information for preclinical studies [1]. A number of imaging tools have been developed for this purpose, such as micro-MRI and optical coherent tomography. Being one of the principal imaging modalities, ultrasound has also been widely investigated for its potential in the field of biomicroscopy. Ever since the 1980s, using high frequency ultrasound to achieve microscopic imaging has been reported, and it is becoming an important imaging tool (sometimes referred to as micro-ultrasound) in many areas such as ophthalmology and dermatology [2], [3].

To facilitate application development in micro-ultrasound, researchers have been devising customized imaging paradigms that extend readily from conventional ultrasound imaging in the 1-10 MHz range. For instance, ultrahigh frame rate retrospective imaging methods has been developed via the use of electrocardiogram as a gating signal [4]. Others have reported the development of adaptive color flow imaging methods that are based on eigen-analysis principles [5]. Realization of these methods in practice, however, is not a straightforward task from an academic research perspective. Although commercial micro-ultrasound scanners have become available in recent years [6], they have been developed specifically for a prescribed set of applications and the embedded system architecture limited their use in research.

Unlike conventional ultrasound, of which various research platforms are readily available on the market [7], the development of micro-ultrasound scanners for research purpose is still in the beginning stage. Some groups have previously attempted to develop their own research-purpose systems for single-element mechanical scanning probes [8]-[11]. Even though they are seemingly programmable systems, their reconfiguration often requires low-level firmware modifications that involve labor-intensive and lengthy coding efforts. To more effectively facilitate research efforts on advanced micro-ultrasound imaging methods, it would be beneficial to develop an open-architecture research platform that is programmable via high-level languages such as C++.

In this work, we present the development of a programmable micro-ultrasound research platform with sufficient processing resources for realizing imaging algorithms that are possibly computation intensive. A system-level overview of this platform will be provided in the next section. Proof-of-concept implementation examples and their processing performance results will also be presented.

II. System Architecture

A. Overall Description

The developed research platform comprises two main blocks: 1) a linear-array scanner front-end, and 2) a programmable PC-based processing back-end, and the two blocks communicate via a parallel port connection. Fig. 1 shows the overall structure of the platform.

During operation, the scanning process first is started by a trigger signal sent to the front-end from the PC. Raw RF data is then acquired and beamformed by the scanning front-end.

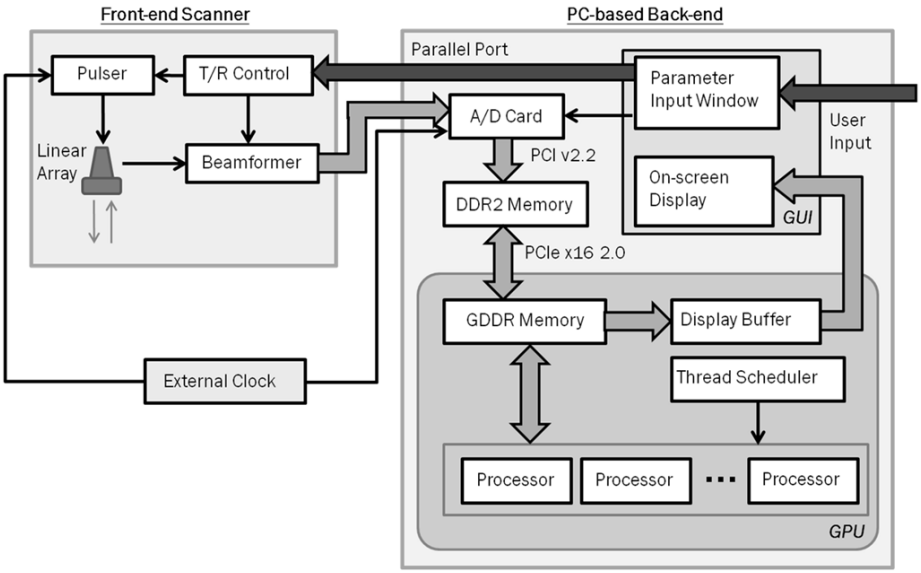

Fig. 1. Overall structure of the programmable micro-ultrsaound research platform. It comprises an array-scanning front-end and a PC-based processing back-end.

TABLE I. CHARACTERISTICS OF THE FRONT-END SCANNER

Parameter	Value
Number of Elements	64
Center Frequency	33 MHz
Kerf Width	14 μm
Pitch	50 μm
Number of Active Channels	32 Eleemnts
Maximum Amplification	70 dB

Sampling of the beamformed data is achieved via a data acquisition card (Gage Applied Tech.) embedded in the PC and the digitized data samples are stored in the PC's random access memory (RAM). After that, the data is streamed into a graphical processing unit (GPU) through the PCIe connection, and the image formation process is carried out. Finally, the resulted image would be displayed on screen in real-time or stored to the PC harddisk memory for further analysis.

B. Front-End Scanner

The front-end scanner is developed previously by Zhang et al. [12] and Table I gives a brief summary of the scanner characteristics. The operations of the scanners are controlled by field programmable gate arrays (FPGA), which in turn can be controlled by the PC-based back-end through the parallel port communication link. A graphical user interface (GUI) has been developed to enable researchers to easily reconfigure system-level parameters like the transmit firing sequence.

C. Back-End Processor

The PC-based back-end of the platform can be divided into three building blocks. Each of these is described briefly below.

1) Data Acquisition Module: Beamformed RF data from the front-end is digitized by a commercial data acquisition card (CS14200, Gage Applied Technologies Inc.). The RF data are sampled at a 200 MHz frequency with 14-bit resolution, and they are streamed to the PC via a PCI link.

2) GPU Data Processing Engine: The GPU of this system (GTX 275; NVIDIA Co.) is responsible for performing post-processing of the acquired RF data. It is housed inside the workstation as an add-on board that is connected through a PCI-Express link. There are several reasons for choosing GPU as the processing core in this platform. First, its higher memory bandwidth in comparison to the PC RAM can help ensure that data access operations would not be a processing bottleneck when processing large RF data streams acquired at a 200 MHz sampling rate (which is necessary because imaging frequencies in the 30 MHz range are used). Second, the multi-core architecture of GPU inherently gives a large parallel processing capacity that may be necessary for realization of computational demanding imaging schemes. Third, since GPU-based parallel processing may be readily controlled through software-based application programming interfaces like CUDA (compute unified device architecture; NVIDIA Co.), it may reduce the development time needed to implement new micro-imaging algorithms as compared to firmware-based modifications.

3) Graphical User Interface: This part of the PC back-end is responsible for displaying images computed from the GPU processor and taking in user inputs. Developers may modify the system parameters online during operations and monitor the changes in the resulted images in real-time.

D. Platform Programmability

In order to facilitate add-on implementation of new micro-imaging algorithms, the software for the research platform has been developed using a module-based approach that is built upon a model-view-controller (MVC) framework used in software design. As illustrated in Fig. 2, the software comprises three main blocks (Model, View, Controller), and they interact with each other via wrapper functions so as to maintain their individuality. These software blocks can be interchanged and scaled owing to the well-defined modularity. For example, the coordinate system in scan conversion can be modified in the View block without affecting the other blocks.

Implementation of new imaging algorithms is achieved via programming of the Model block, which contains the image

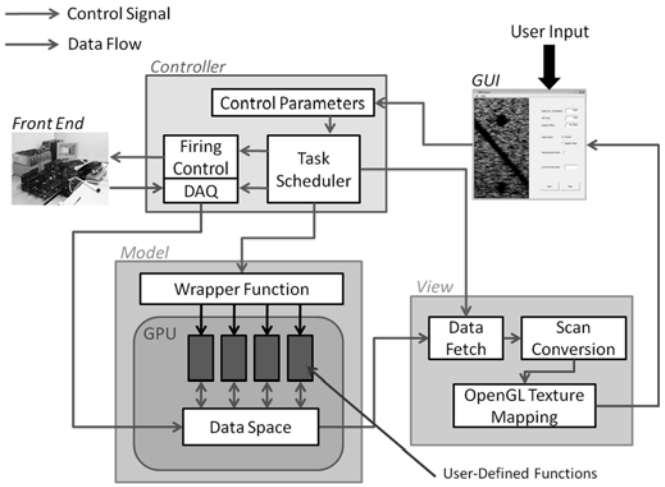

Fig. 2. Conceptual illustration of the module-based software design approach for the research platform. The software is built upon an MVC framework.

TABLE I. IMAGING PARAMETERS FOR THE TEST DATASETS

Parameter	Value
Center Frequency	35 MHz
Pulse Repetition Frequency	B-mode: 3.3 kHz Color flow: 1 kHz
Sampling Frequency	200 MHz
Transmit Focus	8 mm
Maximum Imaging Depth	10 mm
Slow-Time Ensemble Size	B-mode: N/A Color flow: 8
Flow Speed	7 mm/s

data and carries out the image formation process on the GPU. Developers can customize the computational operations on the GPU according to the designed imaging algorithms, and multi-thread parallel processing on the GPU can be invoked through CUDA.

Besides the adoption of a MVC framework for the research platform's software, object-oriented programming has been adopted to handle various processes such as communicating with the data acquisition card and defining various GUI widgets. Such a coding practice is aimed at improving the code-level clarity and modularity for others who wish to develop new imaging algorithms on this platform.

III. TRIAL IMPLEMENTATIONS

A. Implementation Overview

To demonstrate the programmability of our platform, two imaging schemes have been implemented as a proof of concept. The first one is B-mode imaging. This aims at verifying the data flow and the real-time rendering functionality of the system. The second one is an adaptive color flow imaging scheme that uses a third-order down-mixing polynomial filter as the clutter rejection strategy. It serves as an evaluation of the platform's capability in realizing computational demanding algorithms.

B. Performance Analysis Procedure

The performance of the two implemented micro-imaging modes was tested on the PC back-end using simulated micro-ultrasound RF data as the test input. The data was synthesized using the Field-II software, and its imaging view encompasses

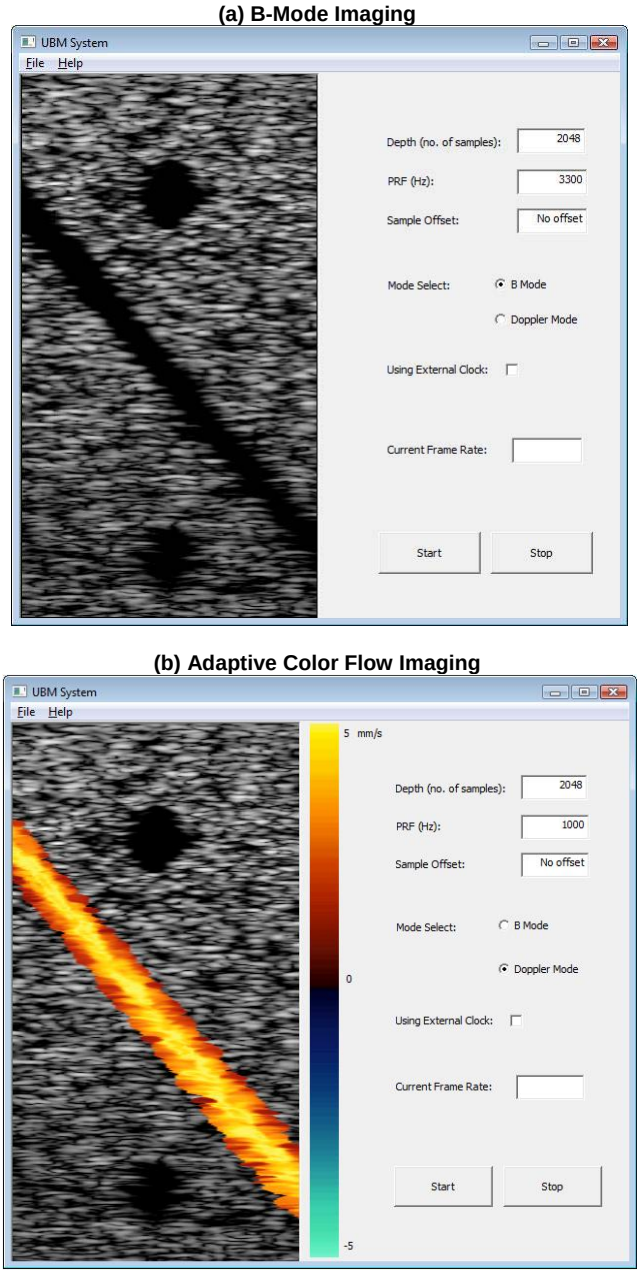

Fig. 3. GUI screenshots of the research platform operating in: (a) B-mode imaging mode, (b) adaptive color flow imaging mode. The imaging view of this example encompasses a 45°-angled flow tube and two cysts (test data are simulated in Field II).

a 45°-angled flow tube and two circular cysts. The imaging parameters used to generate these test datasets are listed in Table II. Note that they are based upon the specifications of the front-end scanner.

C. Results: B-Mode Imaging

Fig. 3a shows a representative screenshot of the research platform's GUI when performing B-mode imaging. The test data were streamed into the PC back-end on a frame-by-frame basis, and the B-mode images were displayed on-screen through continuously rendering by OpenGL. The image size is 2048 × 96 pixels for a given view depth of 10 mm. As can be seen, the imaging view gives an in-plane visualization of the

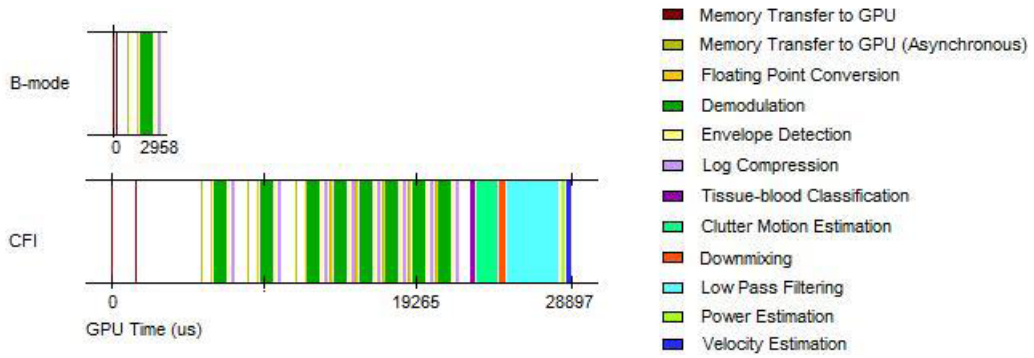

Fig. 4. Single-frame processing time analysis for B-mode imaging and color flow imaging on our research platform. The legend indicates various steps taken in the imaging mode, and their execution time is denoted by their respective color bar in the timing diagram (obtained from CUDA's visual profiler).

flow tube as well as the two circular cysts above and below the flow tube. The lateral resolution is seemingly poorer at greater depths away from the transmit focus. Note that the frame rate is not shown in this figure because the GUI was frozen when the image was captured.

D. Results: Adaptive Color Flow Imaging

Fig. 3b shows the appearance of the research platform's GUI when it is operating in the color flow imaging mode. The shown image is a duplex display whose color gain is linked to the post-filter Doppler power. It can be seen that our custom implementation of color flow imaging has produced a consistent visualization of the parabolic flow profile inside the 45°-angled flow tube, and there is no flashing artifact outside the flow tube region. This indicates that the clutter filtering approach used in this trial implementation is effective in suppressing slow-time clutter in this flow imaging scenario.

E. Results: Processing Time Analysis

Fig. 4 shows the single-frame processing time diagram for the two imaging modes implemented. As can be seen, the time required for computing one frame of B-mode image is near 3000 μs including memory and hardware initiation processes. In contrast, owing to the increased computational complexity, adaptive color flow imaging requires roughly 10 times longer to complete one image frame (i.e. around 30000 μs). If the initialization processes are excluded (as should be the case since they are only executed once during run-time), the platform's processing frame rate for the two imaging schemes are respectively found to be 550 fps (B-mode) and 46 fps (color flow imaging). These are well into the real-time range, thereby demonstrating our research platform's computational capacity that is facilitated through the use of GPU as the processing core.

IV. Conclusion

The research platform has been developed with the intent of facilitating practical realization of various novel micro-ultrasound imaging methods. The module-based programming approach for this platform allows developers to configure the system easily to experiment with various custom micro-imaging algorithms. As demonstrated in the proof-of-concept implementations, realization of computational demanding algorithms is possible given the use of GPU for data processing in the research platform. It is hoped that this platform would be beneficial to micro-ultrasound research and can foster innovative uses of micro-ultrasound in biomedicine.

Acknolwedgements

We are grateful to Prof. Paul Cheung, Mr. Ivan Tsang, and Mr. Donald Chan at the University of Hong Kong for their enthusiastic support on the project.

References

[1] L. Sun, C. L. Lien, X. Xu, and K. K. Shung, "In vivo cardiac imaging of adult zebrafish using high frequency ultrasound (45-75 MHz)", *Ultrasound Med. Biol.*, vol. 34, pp. 31-39, 2008.

[2] F. S. Foster, "Micro-ultrasound takes off (In the biological sciences)," in *Proc. IEEE Ultrason. Symp.*, pp. 120-125, 2008.

[3] F. S. Foster, C. J. Pavlin, K. A. Harasiewicz, D. A. Christopher, and D. H. Turnbull, "Advances in ultrasound biomicroscopy", *Ultrasound Med. Biol.*, vol. 26, pp. 1-27, 2000.

[4] E. Cherin, R. Williams, A. Needles, G. Liu, C. White, A. S. Brown, Y. Q. Zhou, and F. S. Foster, "Ultrahigh frame rate retrospective ultrasound microimaging and blood flow visualization in mice in vivo", *Ultrasound Med. Biol.*, vol. 32, pp. 683-691, 2006.

[5] D. E. Kruse and K. W. Ferrara, "A new high resolution color flow system using an eigendecomposition-based adaptive filter for clutter rejection", *IEEE Trans. Ultrason. Ferroelec. Freq. Contr.*, vol. 49, pp. 1384-1399, 2002.

[6] F. S. Foster, J. Mehi, M. Lukacs, D. Hirson, C. White, C. Chaggares, and A. Needles, "A new 15-50 MHz array-based micro-ultrasound scanner for preclinical imaging", *Ultrasound Med. Biol.*, vol. 35, pp. 1700-1708, 2009.

[7] T. Wilson, J. Zagzebski, T. Varghese, Q. Chen, and M. Rao, "The Ultrasonix 500RP: a commercial ultrasound research interface", *IEEE Trans. Ultrason. Ferroelec. Freq. Contr.*, vol. 53, pp. 1772-1782, 2006.

[8] L. Sun, X. Xu, W. D. Richard, C. Feng, J. A. Johnson, and K. K. Shung, "A high-frame rate duplex ultrasound biomicroscopy for small animal imaging in vivo", *IEEE Trans. Biomed. Eng.*, vol. 55, pp. 2039-2049, 2008.

[9] M. Lewandowski and A. Nowicki, "High frequency coded imaging system with RF software signal processing", *IEEE Trans. Ultrason. Ferroelec. Freq. Contr.*, vol. 55, pp. 1878-1882, 2008.

[10] A. K. Reddy, S. Madala, A. D. Jones, W. A. Caro, J. F. Eberth, T. T. Pham, G. E. Taffet, and C. J. Hartley, "Multichannel pulsed Doppler signal processing for vascular measurements in mice", *Ultrasound Med. Biol.*, vol. 35, pp. 2042-2054, 2009.

[11] M. R. Bosisio, J. M. Hasquenoph, L. Sandrin, P. Laugier, S. L. Bridal, and S. Yon, "Real-time chirp-coded imaging with a programmable ultrasound biomicroscope", *IEEE Trans. Biomed. Eng.*, vol. 57, pp. 654-664, 2010.

[12] L. Zhang, X. Xu, C. Hu, L. Sun, J. T. Yen, J. M. Cannata, and K. K. Shung, "A high-frequency high frame rate duplex ultrasound linear array imaging system for small animal imaging", *IEEE Trans. Ultrason. Ferroelec. Freq. Contr.*, vol. 57, pp. 1548-1557, 2010.