Title	Medical ultrasound imaging: To GPU or not to GPU?
Author(s)	So, HKH; Chen, J; Yiu, BYS; Yu, ACH
Citation	IEEE Micro, 2011, v. 31 n. 5, p. 54-65
Issued Date	2011
URL	http://hdl.handle.net/10722/139257
Rights	©2011 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

Medical Ultrasound Imaging: To GPU or Not to GPU?

Medical ultrasound imaging stands out from other modalities in providing real-time diagnostic capability at an affordable price while being physically portable. This article explores the suitability of using GPUs as the primary signal and image processors for future medical ultrasound imaging systems. A case study on synthetic aperture (SA) imaging illustrates the promise of using high-performance GPUs in such systems.

Hayden K.-H. So
Junying Chen
Billy Y.S. Yiu
Alfred C.H. Yu
University of Hong Kong

Since its introduction in the latter half of the 20th century, ultrasound has enjoyed a unique place in medical-imaging practice.[1] Perhaps most well-known for its ability to scan a developing fetus inside a mother's womb, this imaging modality stands out from others such as computed tomography and magnetic resonance imaging (MRI) in terms of real-time applicability and cost-effectiveness.[2,3] An ultrasound system's compact size is another distinguishing feature of this modality. Modern-day ultrasound scanners are small enough to fit within a rollable trolley or even a portable tablet device.[4] Such portability has made ultrasound indispensable for on-field medical assessments, emergency response, and other agile applications.[5]

Unfortunately, the image quality of existing ultrasound scanners isn't as high as that for computed tomography and MRI. To address this issue, the ultrasound community has developed advanced imaging algorithms such as synthetic aperture (SA) imaging,[6] plane wave compounding,[7] and adaptive beamforming.[8,9] These algorithms can improve focusing power and image contrast while, in the case of SA and plane wave, shortening the data-acquisition period. Researchers have also proposed advanced image-processing techniques such as motion artifact correction[10,11] and 3D image reconstruction to improve the visualization of ultrasound images.[12,13]

Despite the promised image quality, these advanced algorithms demand orders-of-magnitude higher computational capabilities to sustain real-time performance. Moreover, rapid prototyping of these algorithms, whether for research or commercial considerations, is best carried out on software-based general-purpose ultrasound imaging systems instead of dedicated systems. Providing such computational power and high implementation flexibility while maintaining the low power, high portability, and low cost offered in existing systems has made the design of next-generation ultrasound scanners a major challenge for computer system architects.

With countless success stories in using GPUs for application acceleration across many application domains, it's not difficult to imagine that GPUs could also be the ideal solution to address the computational requirements for next-generation ultrasound

Published by the IEEE Computer Society

imaging systems. However, the challenges concerning the use of GPUs in ultrasound systems are beyond simple performance considerations. In this article, we consider GPUs from a range of market segments to explore their viability as a primary computing platform for high-performance bedside ultrasound scanners as well as portable ultrasound systems. We compare these GPUs' power and energy efficiencies and cost-effectiveness to their corresponding hosting CPUs. In addition, we look beyond single GPU systems and study the power-performance trade-offs of multi-GPU and hybrid CPU-GPU systems. Finally, we examine the reasons why GPUs alone might not be adequate to meet all the requirements of next-generation ultrasound systems, and we propose the use of hybrid GPU and field-programmable gate array (FPGA) systems as a possible solution.

Ultrasound imaging using GPUs

By using GPUs as accelerators, researchers across a range of application domains have routinely reported orders-of-magnitude speedup over even the fastest CPUs.[14-16] Indeed, taking advantage of the large number of parallel executing cores in modern GPUs, several recent works have already demonstrated the performance advantages of GPUs in ultrasound image formation when compared to CPU implementations. For example, when using a GPU to implement an SA imaging system's signal-processing stages, D. Romero et al. achieved 1 order-of-magnitude acceleration over their CPU implementation.[17] Similarly, L.-W. Chang, K.-H. Hsu, and P.-C. Li used a GPU to perform color Doppler flow imaging.[18] Furthermore, B.Y.S. Yiu, I.K.H. Tsang, and A.C.H. Yu realized a very high-frame-rate SA image formation at more than 3,000 frames per second (fps) using a three-GPU system.[19]

However, next-generation ultrasound imaging systems are not only computationally demanding. They must also be low power, portable, and low cost, as well as flexible and easy to program, in order to fit in with point-of-care diagnoses, for which ultrasound is clinically poised. Incidentally, the latest developments in GPUs have also made them prime candidates in this regard. In terms of power and portability, GPUs designed for submobile-class computers that can perform 3D rendering at very low power are readily available on the market. In terms of cost, the commoditization of GPUs has made them affordable, especially considering the performance they deliver. In terms of flexibility and programmability, easy-to-use design languages and environments such as CUDA and OpenCL are already available and widely supported by the industry. But, despite these promising characteristics, there has been very little prior work that systematically studies the power and cost efficiency of GPUs and their performance trade-offs, which we focus on in the remainder of this article.

SA imaging case study

To begin our GPU power-performance analysis, we turn to the design of a high-frame-rate SA imaging system as a case study. We chose SA imaging because, if efficiently implemented, it can enhance the quality of existing ultrasound images significantly while also increasing the maximum data-acquisition frame rate. From an architectural-study viewpoint, the SA core algorithm is also interesting because it's highly data parallel and can easily be pipelined, making its implementation on machines with diverse computational and I/O capabilities possible. Thus, we can obtain a bird's-eye view of the GPU computing landscape to date.

Implementation platforms

We implemented the SA core algorithm on five computing platforms (Table 1). We constructed all of them using available commercial-off-the-shelf components. This lets us draw conclusions regarding not only system performance but also the economical viability of deploying such systems.

Platforms 1 to 3 were typical computer systems targeting the high-performance (platform 1), mobile (platform 2), and submobile (platform 3) markets. In particular, platform 1 was a typical high-performance desktop computer powered by a 3.06-GHz Intel Core-i7 quad-core processor with a high-performance Nvidia GeForce GTX 480 GPU. Platform 2 was a notebook

Table 1. Target implementation platforms.

Platform	GPU	GPU memory	CPU	CPU memory	Cost
1	Nvidia GeForce GTX 480	GDDR5, 1.5 Gbytes	Intel Core i7-950 at 3.06 GHz	DDR3, 6 Gbytes	$1,460
2	Nvidia GeForce 9600M GT	GDDR3, 512 Mbytes	Intel Core 2 Duo P8400 at 2.26 GHz	DDR2, 4 Gbytes	$1,360
3	Nvidia ION 2	DDR3, 512 Mbytes	Intel Atom D525 at 1.8 GHz	DDR2, 2 Gbytes	$269
4	(2×GTX 480) + (1×GTX 470)	GDDR5, 4.25 Gbytes	Intel Core i7-950 at 3.06 GHz	DDR3, 6 Gbytes	$2,308
5	2×GTX 480	GDDR5, 3 Gbytes	Intel Core i7-950 at 3.06 GHz	DDR3, 6 Gbytes	$1,959

* DDR: Double Data Rate; GDDR: Graphics Double Data Rate.

computer with an Intel Core 2 Duo P8400 CPU and an onboard Nvidia GeForce 9600M GT GPU.

Sometimes referred to as a nettop, platform 3 represents a new class of submobile computers that consume very low power yet are powerful enough to serve as home theater PCs. In our tests, we used a Zotac ZBOX ID41 running on an Intel 1.8-GHz Atom D525 dual-core processor with an Nvidia ION2 GPU platform.

Platforms 4 and 5 were extensions of platform 1. In platform 4, we connected two GTX 480 GPUs and one GTX 470 GPU to the host CPU for multi-GPU processing. We used platform 5 to perform GPU-CPU hybrid processing. In contrast to platform 4, only two GTX 480s were connected, and the host CPU took up the computational task of the third GPU.

On each platform, we implemented a GPU version of the algorithm using Nvidia CUDA toolkit version 3.0. We also implemented an optimized C version of the same algorithm using the corresponding host CPU. We considered two cases. First, we applied optimizations such as the use of stream SIMD (single instruction, multiple data) extension (SSE) code on one CPU core. This provided us with a base CPU performance for comparison purposes. Subsequently, we used OpenMP to parallelize the optimized C code so as to execute on all possible cores, resulting in the maximum performance achievable on that particular multicore platform.

SA imaging theory

Before we go through the design process, it's useful to first review the theory behind ultrasound imaging. Figure 1 shows the high-level block diagram of a conventional scan-line-based ultrasound system and that of an SA imaging system. As the figure shows, SA imaging differs significantly from traditional ultrasound imaging in the way the images are formed.

A traditional ultrasound system constructs images using the pulse echoes received sequentially from a single-focus transmitting aperture that sweeps across the array transducer.[20] Many firings are typically necessary to produce one complete image of the field of view. Not only does the lack of multifocusing in transmitting beamlines limit the acquired data's quality, but the time lag between beamlines inevitably introduces image artifacts because of the relative motion between the object and scanner. Also, this beam-sweep imaging strategy makes it difficult for existing scanners to reach ultrahigh frame rates of more than 1,000 fps. Such a high data-acquisition frame rate is necessary for diagnoses of dynamic physiological events such as cardiovascular mechanics, strain, and blood flow.[21]

On the other hand, the SA imaging algorithm fires unfocused pulses from each transmitting aperture sequentially and collects pulse echoes over the entire array transducer.[6] As a result, the algorithm can form low-resolution images (LRIs) of the entire field of view after each firing, and can form high-resolution images (HRIs) through recursive compounding of previously generated LRIs and HRIs. In particular, the algorithm can produce the next HRI in the sequence by adding the latest LRI to the previous HRI and subtracting the earliest LRI from the result.

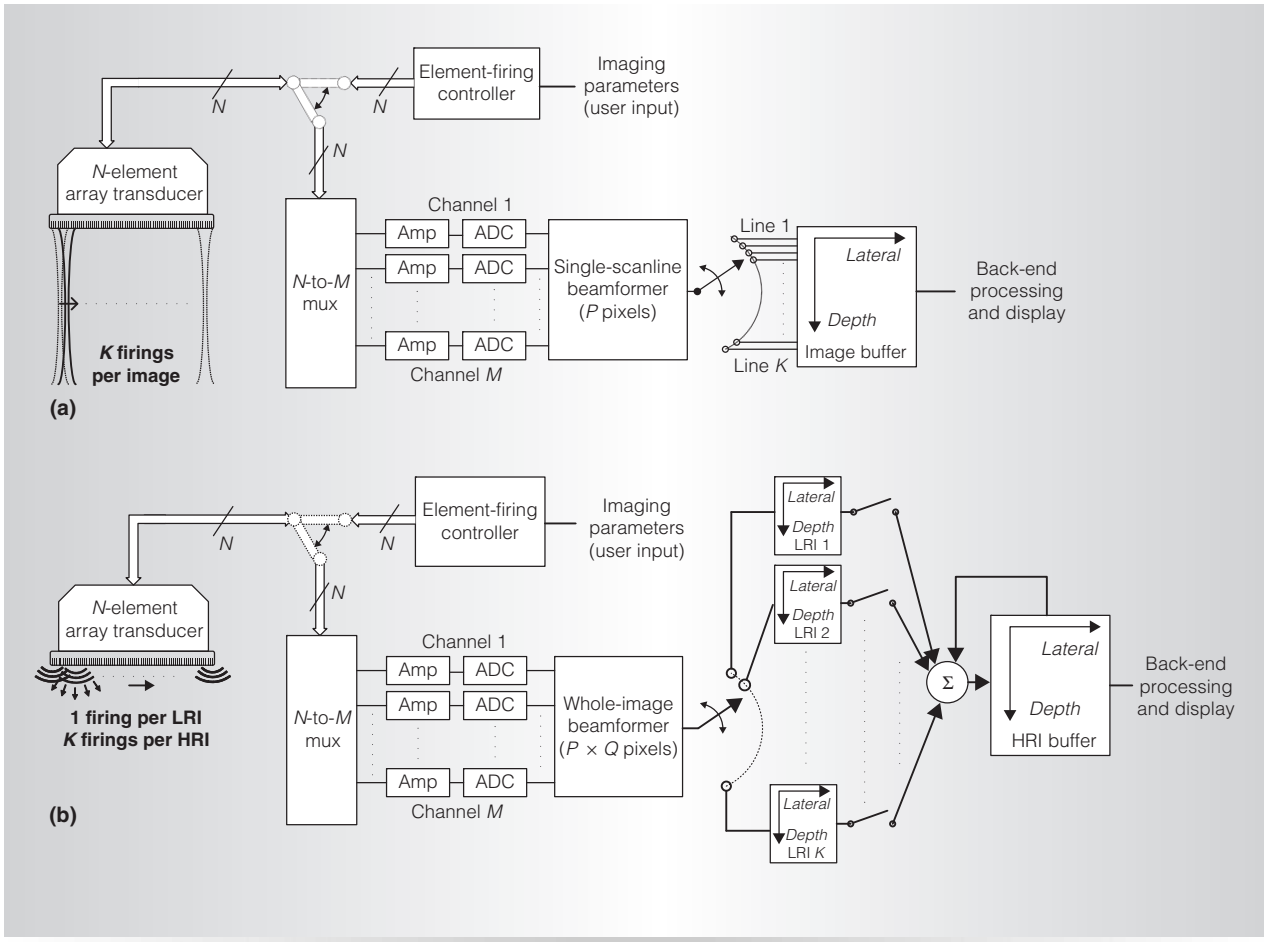

Figure 1. Conventional scan-line-based ultrasound system (a) and synthetic aperture (SA) ultrasound imaging system (b). SA imaging recursively compounds groups of low-resolution images to form a high-resolution image. (ADC: analog-to-digital converter; HRI: high-resolution image; LRI: low-resolution image.)

Besides improving the image quality through compounding, SA imaging makes very high-frame-rate real-time ultrasound imaging possible because each transmission firing can produce an image. However, because all data samples from all receiving elements must be processed at the pulse firing rate, a substantial increase in processing power is required as compared to traditional ultrasound imaging. Note that when compared to the time needed to generate the latest LRI, the time required for HRI recursive compounding is relatively short. Therefore, the LRI generation time is the determining factor for the processing throughput.

GPU implementations

Implementing our SA imaging algorithm on GPU systems primarily involves parallelizing the core algorithm into small independent threads that can be executed by the GPU during runtime. In our work, the imaging process occurs in multiple stages, which follows closely to that detailed by Yiu, Tsang, and Yu.[19] Here, we only highlight the process as shown in Figure 1b.

In the first stage, raw channel data acquired from each transmission firing is assigned a different block of GPU threads to produce an analytic form of the data. Each thread within the thread block is responsible for processing one group of adjacent index positions in the same channel. This allows maximal use of the fast onboard shared memory among the threads within the thread block. The entire resulting analytic data array of this stage is passed to the next stage via onboard global memory.

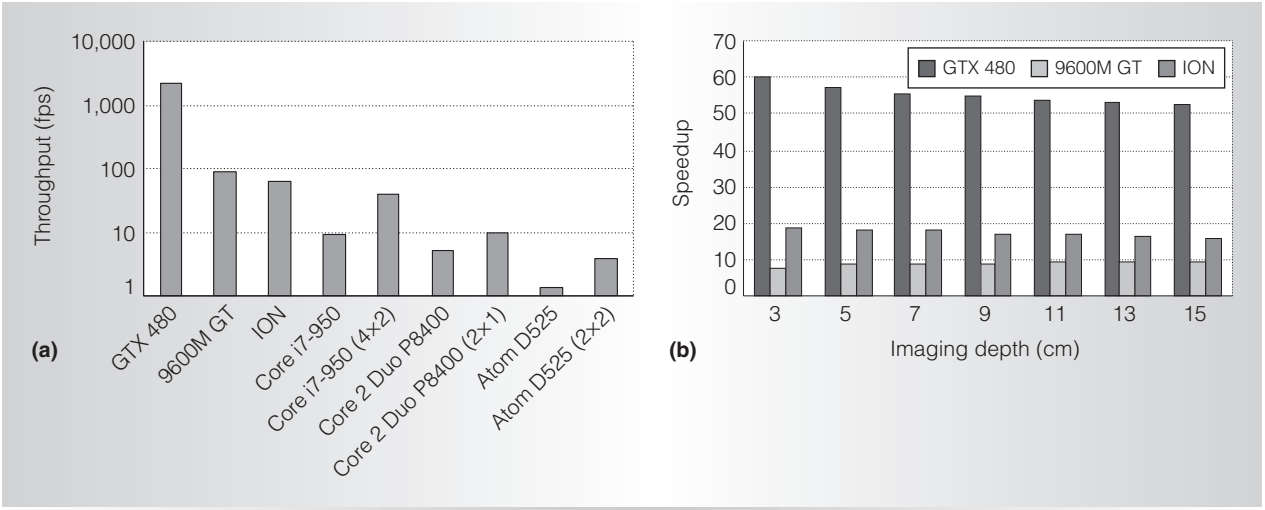

Figure 2. Performance of GPU platforms: throughput in frames per second (fps) (a) and speedup over the host multicore CPU (b). The numbers in parentheses ($c \times t$) indicate that c physical CPU cores were used, each with t hyperthreads. Results are shown for a formation of 512-pixel × 256-pixel images using 128-channel pulse-echo data acquired at a 40-MHz sampling rate. The cross-platform comparison in (a) is for a 9-cm imaging depth.

In a subsequent stage, a delay-and-sum beamforming process is performed among the analytic data from adjacent channels to produce the corresponding LRI scan line. Each thread block is responsible for the computation of one scan line. Each thread within the block, in turn, is responsible for computing one pixel of the scan line. The results are again stored in the global memory before they are processed in the next stage. In the final stage, LRIs from adjacent channels are compounded recursively to produce the final HRI.

The input raw channel data and the output image pixels are stored as 16-bit fixed-point numbers. During the processing stages, we used single-precision floating-point numbers. When compiling our GPU implementations in CUDA, we used fast intrinsic math routines offered by the vendor for better performance at the price of IEEE compliance and, in some cases, reduced precision. In the case of platform 1, for instance, using a fast math routine on the GTX 480 offered up to a 1.78× increase in performance while incurring an average absolute difference (AAD) of 0.31 per pixel. Furthermore, when compared to the host CPU implementation, regardless of the use of fast math routines, both GPU implementations exhibited an AAD of less than 2.15 in a 16-bit value. Because such a minor numerical difference has little influence on the final output ultrasound images' quality, we used fast math routines throughout this work for the sake of performance.

Single GPU performance

We first implemented the SA imaging process on the single-GPU platforms 1 through 3. Figure 2a shows the performance of the SA algorithm, as implemented on our platforms. On each platform, we tested three implementations: a GPU-only implementation, a CPU-only implementation with a single core, and a CPU-only implementation using all available cores. We computed the throughput performance by measuring the end-to-end image formation time for each frame, which included the time for analytic signal conversion, beamforming, and HRI summation. On our GPU implementations, the throughput measurements also included the time to transfer the raw channel data from the CPU host memory to the GPU and the time to transfer the resulting image frame back to the host CPU.

As expected, the high-performance GPU outperformed all other implementations by at least an order of magnitude. Also, the relative performance among the GPUs and among the CPUs conforms to what we

expected on the basis of the devices' target market. However, it was surprising that even the ultralow-power ION GPU consistently outperformed even the highest-performance CPU. In fact, the relative performance of the ION platform continued to improve as the problem size increased, shrinking its gap with the mobile GPU.

Perhaps more interesting is the speedup of the GPUs compared to their hosting CPU, since they were engineered for the same market. Figure 2b shows the speedup of the GPU implementations over their corresponding multicore CPU implementations. The result shows that the high-performance GTX 480 GPU outperformed the equally high-performance CPU by more than 50×. In fact, we observed a speedup of almost 200× in some of our earlier implementations when using only one core of the host CPU. The speed advantage, however, decreases as the problem size increases with imaging depth. One reason is that as the input image size grows, the average memory fetch latency also increases because the size of the working set grows beyond that of the texture cache memory.

Interestingly, the ION GPU again demonstrated a superior average performance gain of 17.22× over the attached Atom processor. In contrast, the 9600M GT GPU had only an average of 8.85× speedup over the hosting Core 2 Duo processor.

Power-performance trade-off

Although the GTX 480 had a clear performance advantage, it also consumed the most power. Figure 3 shows the relationship between power and performance in the three single-GPU platforms. Each vertical cluster of data points is a collection of results from executing the SA algorithm at different imaging depths when using a particular computing setup. The dashed lines at the 33-ms and 1-ms processing times represent the required performance to support basic-video-frame-rate and high-frame-rate SA imaging, respectively. Similarly, the dotted line at the 70-W power consumption roughly determines whether a system can reasonably be employed as a portable ultrasound system.

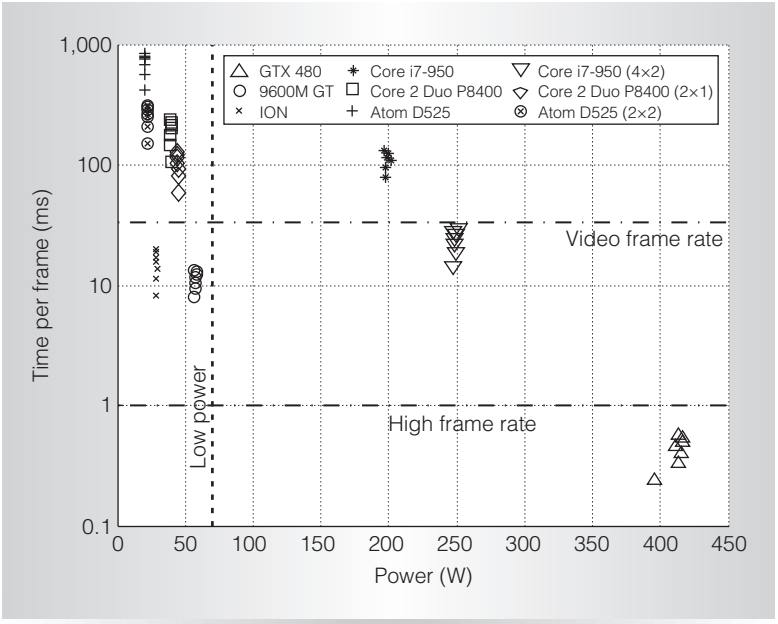

Figure 3. Power-performance trade-off on single-GPU platforms 1 through 3 in Table 1. On each platform, using the host CPU and the attached GPU, we benchmarked three implementations: a GPU-accelerated version, a single-core CPU version, and a multicore CPU version. The data points within each vertical cluster are the computing results from different imaging depths.

Among the tested combinations, only the two low-power GPU systems achieved a video-processing frame rate at relatively low power. Furthermore, only the GTX 480 system was capable of performing high-frame-rate SA imaging when running at greater than 400-W power consumption.

On the CPU front, although the two mobile CPUs ran at very low power, their performance was relatively poor. On the other hand, although the high-performance Core i7 CPU achieved video frame rates at moderate image depth, it consumed more than 240 W.

Energy efficiency

To understand the trade-off between power and performance, we considered two metrics in our case study: power throughput ratio and energy throughput ratio.

We define the power throughput ratio as

$$\frac{\text{power}}{\text{throughput}} = \frac{\text{energy/second}}{\text{frames/second}} = \frac{\text{energy}}{\text{frame}}$$

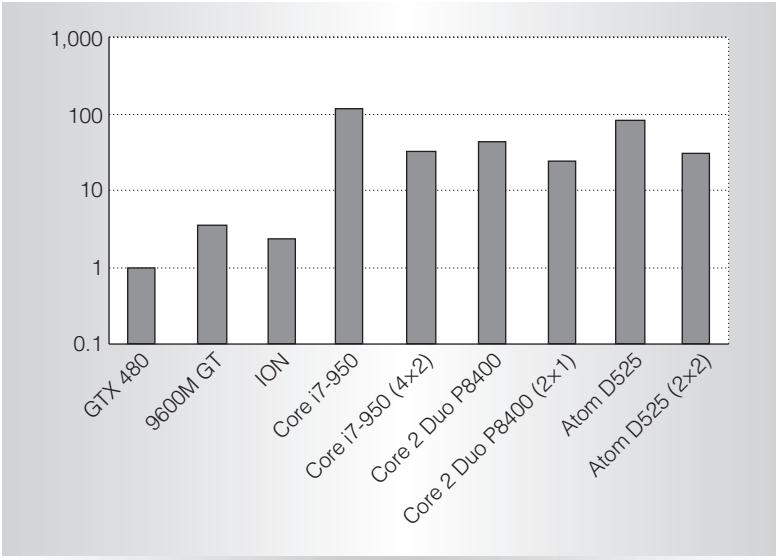

Figure 4. Normalized power throughput ratios (depth = 9 cm, relative to the GTX 480). All three GPUs were at least an order of magnitude more power efficient than their corresponding hosting CPU, mainly due to their superior performance with a marginal increase in power consumption. Using multiple cores in the CPU concurrently had the same net effect in increased power efficiency when compared to their corresponding single-core implementation.

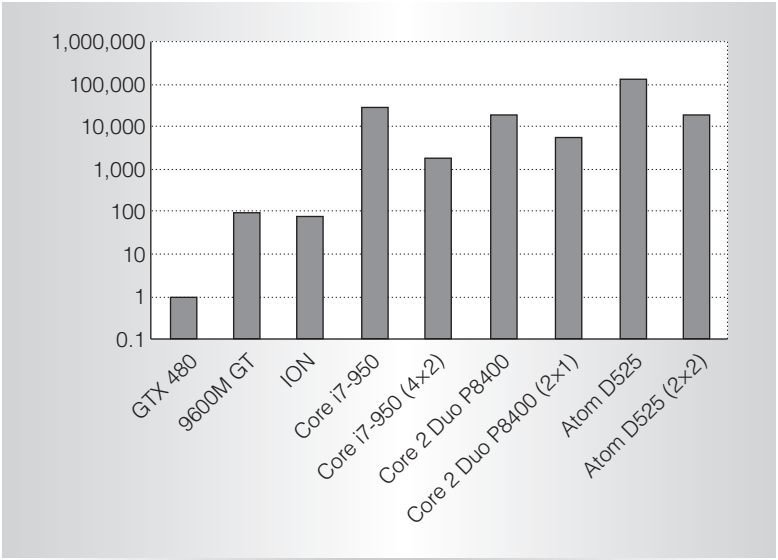

Figure 5. Normalized energy throughput ratios (depth = 9 cm, relative to GTX 480). All three GPUs were at least 2 orders of magnitude more efficient than their corresponding hosting CPUs in terms of energy throughput ratios (ETRs). The relatively poor throughput performance of the Atom CPU made it trail behind its CPU counterparts when performance was taken into account in ETR measurements.

It measures a system's power efficiency and essentially determines the energy required to process one frame of an image. This is particularly important for portable systems because it affects their battery life.

As Figure 4 shows, although the GTX 480 consumed the most power, it was indeed the most power efficient among all systems. In fact, all three GPUs were at least an order-of-magnitude more power efficient than their corresponding host CPUs, with the high-performance Core i7-950 CPU performing the worst. The result shows the clear advantage of the GPU architecture optimized for highly data-parallel applications such as the SA algorithm. For instance, comparing the results of the GTX 480 (415 W, 2,480 fps, 40-nm process, 1.038 V) and the Core i7 CPU (248 W, 45 fps, 45-nm process, 1.375 V max) shows that a 1.7× increase in power consumption results in a 55× increase in performance. Consequently, although the GPU runs at higher power, it finishes processing one image frame far more quickly than the CPU, resulting in a 33× lower energy consumption per frame.

Although power throughput ratios can indicate the relative energy power-performance trade-off among the tested systems, they don't consider the system's actual throughput performance. For that, the energy throughput ratio is a better measurement (see Figure 5).[22]

For the purpose of this work, we define a system's energy throughput ratio as

$$\frac{\text{energy/frame}}{\text{frame/second}} = \frac{\text{power}}{(\text{throughput})^2} \qquad (1)$$

The energy throughput ratio measures a system's energy efficiency, given a particular image-processing frame rate of interest to the designer.

Compared to the power efficiency results in Figure 4, the gaps between GPUs and CPUs are much wider because of the square term of throughput in Equation 1. However, because this measurement considers the actual throughput performance, the high-performance CPU's energy efficiency has become relatively better than its low-power mobile siblings.

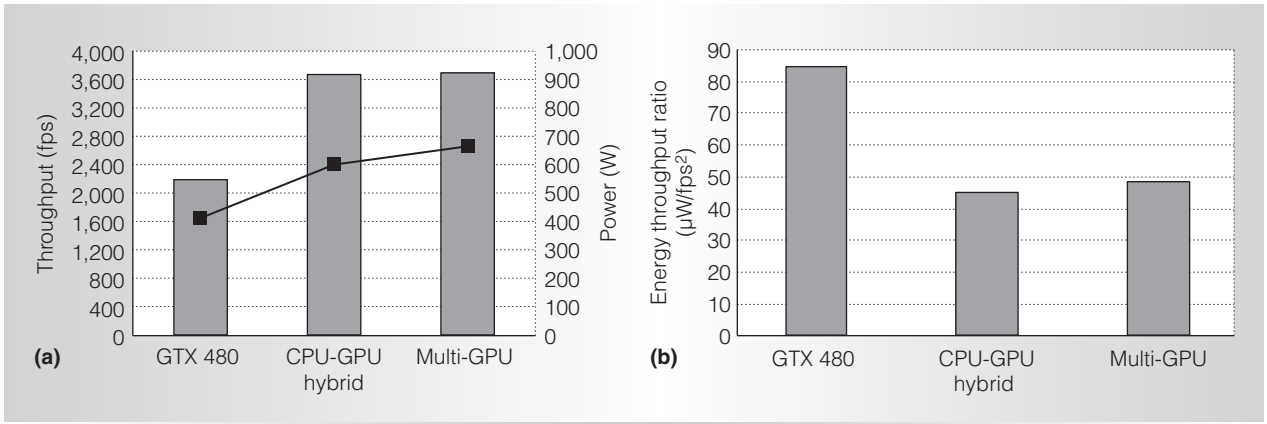

Figure 6. Performance, power, and energy efficiencies of multi-GPU and hybrid CPU-GPU systems compared with a single-GPU accelerated system: performance (columns) and power (line) (a), and energy throughput ratios (b). Compared to the single-GPU system, the additional GPU and CPU provided significant speed improvement with a marginal increase in power consumption, making both of them more energy efficient.

The results from Figures 4 and 5 suggest that system performance is a dominating factor that determines its power and energy efficiency. In the case of GPUs, for example, any increase in the number of processing cores translates directly into an almost linear performance increase. Yet, the associated power consumption doesn't usually increase linearly, because the increased capacitance associated with the extra computational core contributes to only a small portion of the system's overall power consumption. In the case of the GTX 480 and the ION2, both were manufactured using the same 40-nm technology. The GTX 480 has 480 computing cores, whereas the ION2 has 16—a 30:1 ratio. Although running at a slightly different frequency, the GTX 480 was consistently 33× faster than the ION2 across all imaging depths. At the same time, the power consumption of the GTX 480 was only about 14× higher. Because of this incremental difference in power and performance, the GTX 480 was consistently more power and energy efficient.

Multi-GPU and hybrid CPU-GPU implementations

To further enhance performance, we used a multi-GPU (platform 4) and a hybrid CPU-GPU (platform 5) setup to implement our SA algorithm. Platform 4 divided the LRI computations between the two GTX 480s. The GTX 470 then processed the results to generate the final HRIs. In platform 5, the host quad-core CPU took the place of the GTX 470 for HRI computation, and the two GPUs performed LRI computations as before.

Figure 6a shows the performance and power consumption for the multi-GPU and hybrid platforms. The performance of the three-GPU and hybrid CPU-GPU platforms is almost identical, offering a speedup of 1.68× and 1.66× when compared to the performance of the single GTX 480. With three GPUs performing the computation in platform 4, whereas only two GPUs were used in platform 5, a far better performance out of platform 4 might be expected. To understand this apparent discrepancy, we obtained the runtime profiles of both platforms, as shown in Figure 7.

The runtime profile of the three-GPU platform in Figure 7 shows that HRI processing was relatively straightforward compared to LRI computation. As a result, rather than performing useful computation, the extra GTX 470 in platform 4 spent most of its time on data transfer and was otherwise idled. On the other hand, in the case of the hybrid CPU-GPU platform 5, although the CPU's computing capability was lower than the GTX 470, using two CPU cores for concurrent HRI computation enabled it to keep up with the LRI generation speed of the two GPUs. Consequently, we could completely overlap the computation

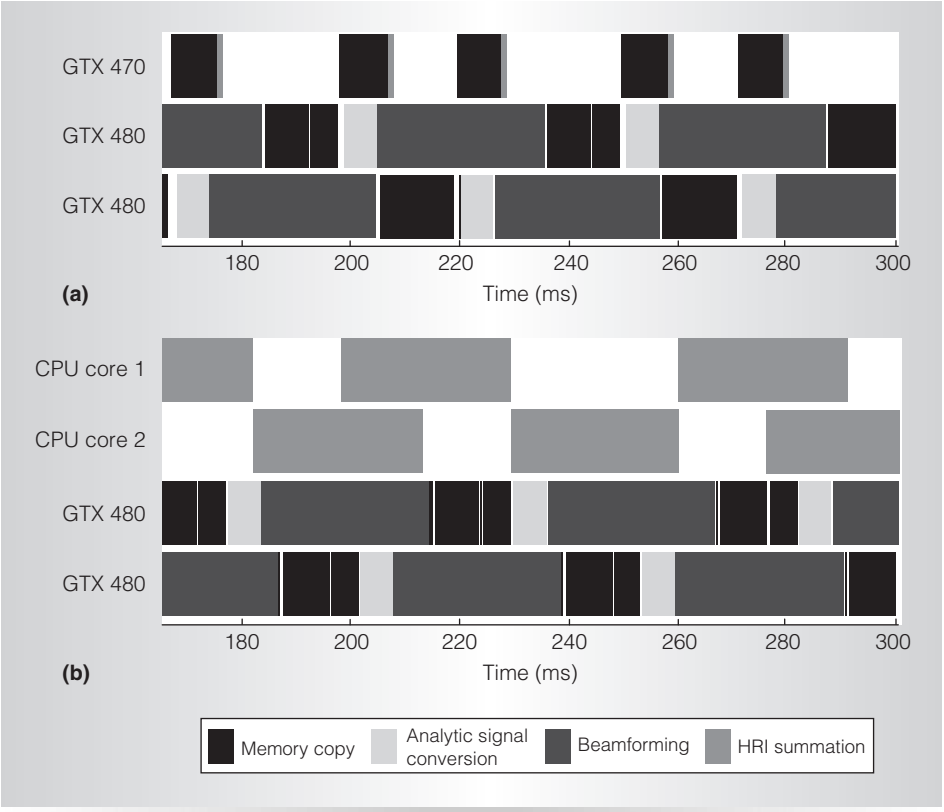

Figure 7. Profile of tasks running on multi-GPU (a) and hybrid CPU-GPU (b) platforms. The GTX 470 in the multi-GPU system was idled most of the time because it consumed LRIs for HRI computation at a much higher rate than the two GTX 480s can produce them. However, on the CPU-GPU platform, although HRI computation on the CPU was slower than on the GTX 470, it was completely overlapped by the LRI computation, making the platform as fast as the multi-GPU platform.

of HRIs with the computation of LRIs in platform 5. Furthermore, unlike the case in platform 4, in which the LRIs must be transferred to the GPU for HRI computation, the CPU cores in platform 5 directly accessed the main memory where the LRIs were residing. This elimination of extra memory transfer significantly reduced contention in the main memory and I/O buses, resulting in an overall far more efficient system. In fact, we had to devote considerable effort in manually tuning the relative schedule of HRI and LRI computations on the three GPUs in platform 4 to avoid main-memory access conflicts that would otherwise have incurred extra delay.

With one less GPU running, the power consumption of platform 5 was about 60 W lower than platform 4 during program execution. The lower power, combined with equally capable performance, made the hybrid CPU-GPU platform the most energy efficient, as Figure 6b shows.

Implementation cost

Finally, we estimated the cost per throughput for all the platforms. Figure 8 shows the results. Despite the high cost for platforms 1, 4, and 5, their superior performance using the high-performance GPUs made them an order-of-magnitude more cost-effective than other platforms. In particular, because it maximized all computing resources, the hybrid CPU-GPU platform was the most cost-effective platform tested.

When GPU is not enough

On the basis of the SA imaging results alone, it might seem that using GPUs is the "magic bullet" for medical ultrasound imaging.

However, SA imaging is only one of many ultrasound algorithms under development. Our own experience in developing these algorithms suggests that using a GPU by itself is inadequate in many cases. For instance, consider the case of block matching for motion estimation, which is useful for vector-flow imaging, elastography, motion artifact correction in compound imaging, and so on. At an imaging depth of 5 cm and with 11 × 11 block search windows, a mere 5 fps was achievable so far with our high-performance GPUs, which is similar to the result that G. Kiss et al. reported.[23]

In another instance, we developed an in-house adaptive beamforming algorithm that enhances image contrast by adaptively adjusting apodization weights for delay-and-sum beamforming. With a 64-channel receiving aperture (M) and a 32-channel subaperture array, only 2 fps was achievable with our high-performance GPU. With an $O(M^3)$ complexity, the performance decreases rapidly with larger receiving apertures in advanced ultrasound transducers.

Another problem lies with data I/O. By shifting the task of image formation to GPUs, we are essentially pushing the GPU's role further toward the ultrasound front end. As such, we must stream the multichannel unprocessed data from the array transducer to the GPU continuously at a very high speed. Assuming a 10-cm imaging depth, a 40-MHz sampling rate, 128 channels, and 2 bytes per sample, each firing generates approximately 1.2 Mbytes of pulse-echo data. Thus, assuming 3,000 firings per second, the data throughput would be 3.7 Gbytes per second. Such bandwidth is far beyond the capability of most commodity I/O standards available today and is already pushing the performance limit of the onboard communication buses to which most GPUs are connected.

To address these issues, we are exploring the use of FPGAs combined with GPUs to form a heterogeneous computing system for ultrasound imaging. In our case, we use FPGAs not only as low-level signal processors but also as part of the high-level computing platform. Besides acting as an interface to the transducer front end, the FPGAs are responsible for preprocessing the data before it's transferred to either the host CPU or the GPUs. In essence, the FPGAs will compute near the data source, whereas the GPUs will compute near the real-time display to the user. The goal of this setup is to reduce as many unnecessary data transfers as possible that would otherwise slow down the overall performance, as in an SA multi-GPU implementation.

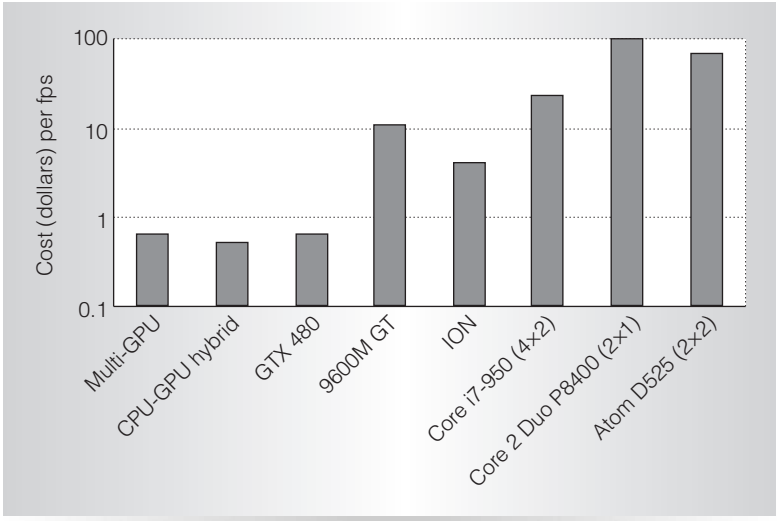

Figure 8. Cost per throughput performance on each platform (measured in processed fps), with and without GPU acceleration. Cost was estimated based on the suggested retail price of the system components. For desktop systems with discrete GPUs, the corresponding number of CPUs and GPUs were tallied with the cost of the supporting platform. In the case of the mobile platforms with onboard GPUs, the entire system cost was used in both the CPU and GPU cost estimation.

From the study on SA imaging, it's clear that using high-performance GPUs in next-generation ultrasound imaging systems is promising. Many highly parallelizable, data-intensive ultrasound imaging algorithms fit very well with the single-instruction, multiple-thread (SIMT) architecture of modern GPUs, giving rise to the resulting systems' exceptional performance. Such orders-of-magnitude higher performance with only moderate additional power consumption and cost further translates into even better power and energy efficiency, as well as cost-effectiveness.

Furthermore, the advantage of GPUs over CPUs extends beyond the high-performance segment into the cost- and power-sensitive

submobile market segment. These new GPUs have demonstrated respectable performance, given their cost and power consumption, making them likely candidates for future portable or entry-level ultrasound systems.

Nevertheless, it's also clear that using GPUs alone is inadequate to address all the processing needs for the most advanced systems, especially when considering the entire image-processing path from the front-end probe to the back-end user display. In such cases, hybrid computing systems that combine CPUs, GPUs, and even FPGAs are more likely to excel. MICRO

Acknowledgments

This work is supported in part by the Hong Kong Innovation and Technology Fund (ITS/492/09 and InP/211/10) and the Research Grants Council of Hong Kong (project GRF 716510).

References

1. P.N.T. Wells, ''Ultrasound Imaging,'' *Physics in Medicine and Biology,* vol. 51, no. 13, 2006, pp. R83-R98.
2. A.B. Wolbarst and W.R. Hendee, ''Evolving and Experimental Technologies in Medical Imaging,'' *Radiology,* vol. 238, no. 1, 2006, pp. 16-39.
3. S.M. Bierig and A. Jones, ''Accuracy and Cost Comparison of Ultrasound Versus Alternative Imaging Modalities, Including CT, MR, PET, and Angiography,'' *J. Diagnostic Medical Sonography,* vol. 25, no. 3, 2009, pp. 138-144.
4. K.E. Thomenius, ''Miniaturization of Ultrasound Scanners,'' *Ultrasound Clinics,* vol. 4, no. 3, 2009, pp. 385-389.
5. C.L. Moore and J.A. Copel, ''Point-of-Care Ultrasonography,'' *New England J. Medicine,* vol. 364, no. 8, 2011, pp. 749-757.
6. J.A. Jensen et al., ''Synthetic Aperture Ultrasound Imaging,'' *Ultrasonics,* vol. 44, supplement 1, 2006, pp. e5-e15.
7. G. Montaldo et al., ''Coherent Plane-Wave Compounding for Very High Frame Rate Ultrasonography and Transient Elastography,'' *IEEE Trans. Ultrasonics, Ferroelectrics and Frequency Control,* vol. 56, no. 3, 2009, pp. 489-506.
8. J.-F. Synnevag, A. Austeng, and S. Holm, ''Adaptive Beamforming Applied to Medical Ultrasound Imaging,'' *IEEE Trans. Ultrasonics, Ferroelectrics and Frequency Control,* vol. 54, no. 8, 2007, pp. 1606-1613.
9. J.-F. Synnevag, A. Austeng, and S. Holm, ''Benefits of Minimum-Variance Beamforming in Medical Ultrasound Imaging,'' *IEEE Trans. Ultrasonics, Ferroelectrics and Frequency Control,* vol. 56, no. 9, 2009, pp. 1868-1879.
10. J.-S. Jeong et al., ''Effects and Limitations of Motion Compensation in Synthetic Aperture Techniques,'' *Proc. IEEE Ultrasonics Symp.,* IEEE Press, 2000, vol. 2, pp. 1759-1762.
11. S.I. Nikolov and J.A. Jensen, ''K-Space Model of Motion Artifacts in Synthetic Transmit Ultrasound Imaging,'' *Proc. IEEE Ultrasonics Symp.,* IEEE Press, 2003, vol. 2, pp. 1824-1828.
12. R.W. Prager et al., ''Three-Dimensional Ultrasound Imaging,'' *J. Eng. in Medicine,* vol. 224, no. 1, 2010, pp. 193-223.
13. K. Karadayi, R. Manuguli, and Y. Kim, ''Three-Dimensional Ultrasound: From Acquisition to Visualization and from Algorithms to Systems,'' *IEEE Rev. Biomedical Eng.,* vol. 2, 2009, pp. 23-39.
14. J. Nickolls and W.J. Dally, ''The GPU Computing Era,'' *IEEE Micro,* vol. 30, no. 2, 2010, pp. 56-69.
15. M. Garland et al., ''Parallel Computing Experiences with CUDA,'' *IEEE Micro,* vol. 28, no. 4, 2008, pp. 13-27.
16. J.D. Owens et al., ''GPU Computing,'' *Proc. IEEE,* vol. 96, no. 5, 2008, pp. 879-899.
17. D. Romero et al., ''Using GPUs for Beamforming Acceleration on SAFT Imaging,'' *Proc. IEEE Int'l Ultrasonics Symp.* (IUS 09), IEEE Press, 2009, pp. 1334-1337.
18. L.-W. Chang, K.-H. Hsu, and P.-C. Li, ''Graphics Processing Unit-Based High-Frame-Rate Color Doppler Ultrasound Processing,'' *IEEE Trans. Ultrasonics, Ferroelectrics and Frequency Control,* vol. 56, no. 9, 2009, pp. 1856-1860.
19. B.Y.S. Yiu, I.K.H. Tsang, and A.C.H. Yu, ''GPU-Based Beamformer: Fast Realization of Plane Wave Compounding and Synthetic Aperture Imaging,'' *IEEE Trans. Ultrasonics, Ferroelectrics and Frequency Control,* vol. 58, no. 8, 2011, pp. 1698-1705.

20. T.A. Whittingham, ''Medical Diagnostic Applications and Sources,'' *Progress in Biophysics and Molecular Biology,* vol. 93, nos. 1-3, 2007, pp. 84-110.
21. S. Wang et al., ''A Composite High-Frame-Rate System for Clinical Cardiovascular Imaging,'' *IEEE Trans. Ultrasonics, Ferroelectrics and Frequency Control,* vol. 55, no. 10, 2008, pp. 2221-2233.
22. T.D. Burd and R.W. Brodersen, *Energy Efficient Microprocessor Design,* Springer, 2002.
23. G. Kiss et al., ''Performance Optimization of Block Matching in 3D Echocardiography,'' *Proc. IEEE Int'l Ultrasonics. Symp.* (IUS 09), IEEE Press, 2009, pp. 1403-1406.

Hayden K.-H. So is an assistant professor in the Department of Electrical and Electronic Engineering at the University of Hong Kong. His research interests include computer architecture, reconfigurable computing, field-programmable gate arrays, GPUs, and operating-system design. So has a PhD in electrical engineering and computer sciences from the University of California, Berkeley. He's a member of IEEE.

Junying Chen is pursuing a PhD in electrical and electronic engineering at the University of Hong Kong. Her research interests include computer architecture and parallel computing, and their applications in ultrasound imaging systems. Chen has a BEng in electronic and information engineering from Zhejiang University. She's a student member of IEEE.

Billy Y.S. Yiu is a research associate in the Medical Engineering Program at the University of Hong Kong. His research focuses on medical ultrasound imaging, specifically the design of advanced imaging paradigms and new system hardware. Yiu has an MPhil in electrical and electronic engineering from the University of Hong Kong. He's a member of IEEE.

Alfred C.H. Yu is a research assistant professor in the Medical Engineering Program at the University of Hong Kong. His research interests include developing new medical ultrasound-imaging methods and systems for clinical applications and studying the fundamentals of various ultrasound-induced wave-matter interactions. Yu has a PhD in electrical engineering from the University of Toronto. He's a member of IEEE.

Direct questions and comments about this article to Hayden K.-H. So, Room 516, Chow Yei Ching Building, Dept. of Electrical and Electronic Engineering, University of Hong Kong, Hong Kong; hso@eee.hku.hk.