Researchers at the Advanced Digital Sciences Center (ADSC) have obtained notable success in enabling software engineers to produce hardware designs with the quality expected of experienced hardware designers, but with the fast design cycle of high-level languages and little to no need for hardware design expertise. To date, ADSC researchers have demonstrated the generation of good-quality FPGA hardware designs from high-level C-like code intended for GPU programming, with minimal hints from the programmer to guide the design process. The resulting reduction of hardware design time from months down to minutes will radically enhance a software team’s ability to innovate rapidly and make iterative improvements to algorithms.

In addition to its broader value to the computing community, this work is benefiting ADSC internally. ADSC’s audiovisual researchers now can quickly synthesize hardware for their digital signal processing algorithms, which are a vital ingredient in ADSC’s broader research program to provide low-cost, high-quality, real-time immersive telepresence, a grand challenge in itself [NAE11]. A video about this work is available through the ADSC web site, http://adsc.illinois.edu.

1 The Need for Hardware Synthesis Tools That Software Engineers Can Use

Although Moore’s law predicts a doubling in manufacturable transistor density every 18 months, the ability of hardware designers to effectively use those transistors is growing much more slowly. Thus during decades of research, significant effort has been expended on developing hardware synthesis tools that can efficiently turn high-level programming language code into high performance, power efficient hardware designs [MS09]. However, designers must still possess significant hardware experience and expend considerable effort to annotate and transform high level language code for hardware synthesis tool use. Further, to make the synthesis process tractable, many tools still require specialized languages or language subsets and restrictive coding styles. Thus prior work in high-level synthesis (HLS) primarily targets a user base of experienced hardware designers who want to reduce design time. Now ADSC has added to the challenge by demanding that high-level synthesis produce good quality designs even in the hands of users who know little or nothing about hardware design.

Driven by the need for power-efficient application acceleration, computing hardware platform design is exhibiting a pervasive trend toward heterogeneous hardware, i.e., combining CPUs, graphical processing units (GPUs) and field-programmable gate arrays (FPGAs) in a single system. Though much faster than for a full-custom design, the design time for FPGA platforms remains orders of magnitudes longer than equivalent development in a high-level language. This is because a software engineer must first develop the algorithm, then work with a hardware engineer who has expertise in FPGA design. The hardware portion of the design cycle can easily last six months, by which time the software engineer may have thought of many improvements to the original algorithm that are not reflected in the FPGA design. Further, perhaps a slightly different version of the algorithm would have produced a better FPGA design, but the potential improvement will not generally be apparent to the team until the end of the design cycle. Thus even with FPGAs, the long design cycle inhibits the team’s ability to innovate rapidly and to make iterative design improvements.

From our point of view, the choice of hardware engineers as the user base is a critical, fundamental flaw in the design and target of current HLS tools. A much larger, yet more challenging, market is represented
by software engineers with little or no hardware expertise, but who nevertheless demand hardware acceleration. Thus, the fundamental difference between ADSC’s approach and the approach of other HLS tools is the explicit target of usability by software designers with no hardware design experience, and potentially little software design experience.

2 ADSC Contributions
ADSC’s high-level synthesis research has made great strides towards its three fundamental goals: ease of use by software engineers, automated mapping to hardware, and automated optimization of source code.

Uniform, easy programming interface. Due to the long-standing dominance of C/C++ programming languages, and the emergence of popular C-style GPU programming languages exemplified by CUDA and OpenCL, ADSC selected these languages as the primary input for HLS. Our HLS tool, FCUDA, was the first to generate hardware from a GPU language (CUDA). FCUDA’s hardware designs achieve performance on par with the GPU source code that they are derived from, but consume 90% less power. This shows that high parallel computing tasks expressed in GPU languages can be translated into customized parallel computing tasks on FPGAs in a manner that provides high performance and low power consumption; the significance of this feat is demonstrated by the two best paper prizes that the papers generated by the FCUDA project have won. Thus FCUDA provides engineering teams with a midpoint in performance between a pure software implementation and a manual FPGA hardware implementation, but with little to no additional design effort or design time to generate the hardware.

FCUDA also allows software engineers to use a single language to program heterogeneous computing platforms that include both GPUs and FPGAs, which was not possible in the past. This new programming paradigm can support high performance portability across GPUs and FPGAs and take advantage of both types of accelerators, thus targeting different applications seamlessly.

Automatic mapping to hardware. HLS tools traditionally require manual insertion of annotations that provide guidance in producing the hardware design. Our first paper in the FCUDA project achieved FPGA performance on a par with that of CUDA application code, but still required manual annotation effort [PGS09]. Our second paper in the FCUDA project, which received the best paper award at the top conference for research on reconfigurable hardware, presented an effective design space exploration technique that eliminates the need for many manual annotations, thereby reducing design effort and shortening the design cycle, but without reducing performance [PLSG11]. Our current work eliminates the need for most annotations, thereby further reducing design effort, and without compromising on performance [PLRC12].

Figure 1 - FCUDA II automatically explores a variety of solutions (left) that trade between hardware area consumed (X axis) and computation latency (Y axis). FCUDA II then selects the design that gives performance parity on average with GPU implementations (center) but with average 90% lower energy consumption (right).

We carried out a systematic study of the productivity, performance, and software constraints of HLS, using ADSC’s work on stereo matching of color video as the driver application, as well as some common computing kernels and security-related application algorithms. This is a systematic and unbiased study of
the usability of HLS with code written by software designers who have no hardware knowledge [RLL11a, RLL11b]. For the stereo matching algorithms, through a series of high-level transformations and optimizations together with the HLS tool, we demonstrated between 3.5X and 67.9X speedup over software (which is less than achievable by manual RTL design), with a five-fold reduction in design effort compared to manual hardware design. This project exposed unique opportunities to improve HLS tools to further reduce the performance gap between HLS and manual RTL design, which we will pursue in the future.

**Automated optimization of source code.** Currently, software designers must modify or entirely rewrite GPU source code to maintain good performance when they switch to a different target implementation platform. ADSC is working to produce techniques to automatically optimize GPU source code, so that the code can be automatically customized for multiple GPU platforms or as a preparation step before HLS. Our most recent work shows how to efficiently explore the multi-dimensional design space associated with certain aspects of platform customization [LCR12, CLRC12]. In particular, to reduce user effort, we have determined how to optimize GPU code branch divergence and memory access coalescing, which are critical aspects of GPU execution performance. Automatic optimization to reduce the impact of these factors is a major factor in providing performance portability across multiple GPU platforms. GPU performance heavily depends on computation throughput – ensuring that the GPU’s memory bandwidth and computation resources are highly utilized. Code branch divergence and uncoalesced memory accesses are two critical cases that reduce compute resource utilization and memory bandwidth utilization, respectively. However, due to the large number of independent threads during execution on GPUs, it is difficult or impossible to entirely eliminate branch divergence or uncoalesced memory accesses. Furthermore, optimizations for branch divergence may harm memory access coalescing and vice versa. We developed an effective metric for estimating the performance impact of branch divergence, and showed that new thread reorganization mechanisms that use our metric improve performance by up to 2.3X, compared to the original kernel.

**About Us**
The **Advanced Digital Sciences Center** (ADSC), established in 2009 and located in Singapore, is a wholly-owned subsidiary of the University of Illinois at Urbana-Champaign. ADSC’s research program in interactive digital media is funded by the Agency for Science, Technology, and Research (A*STAR).

**Deming Chen** is an associate professor in the Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. He received an Arnold O. Beckman Research Award, NSF CAREER Award, ACM SIGDA Outstanding New Faculty Award, and four best paper awards. He is an associate editor for IEEE Transactions on Very Large Scale Integration Systems and IEEE Transactions on Circuits and Systems I. Deming received his Ph.D. from the University of California at Los Angeles in 2005.

**Eric Liang** is a postdoctoral fellow at ADSC. He received the Best Paper Award from FCCM 2011 and a Best Paper Award nomination from CODES+ISSS 2008. He is also a recipient of the National University of Singapore Dean's Graduate Research Excellence Award. Eric received his Ph.D. from the National University of Singapore in 2010.
Kyle Rupnow is a research scientist at ADSC. He previously received a Sandia National Laboratories Excellence in Engineering Fellowship and an NSF Graduate Research Fellowship honorable mention. He is also a recipient of the Gerald Holdridge award for Tutorial Development and the University of Wisconsin-Madison Capstone Ph.D. teaching award. Kyle received his Ph.D. from the University of Wisconsin – Madison in 2010.

Zheng Cui is a software engineer at ADSC. She received her B.Comp (Honours) in computational biology from the National University of Singapore in 2011. She is now working with Dr Eric Liang and Dr Kyle Rupnow on research projects related to graphics processors.

References