GPU-Accelerated Computing: Maximizing Performance for the 24/7 Semiconductor Manufacturing Environment

By Aki Fujimura, CEO of D2S, Inc.

2016

Executive Summary

The computing applications used in semiconductor design and manufacturing have ever-increasing requirements for speed, accuracy and reliability. The continuation of Moore's Law creates a perpetual demand for greater accuracy as, with each new process node, larger numbers of increasingly smaller features are crowded onto each mask and wafer. Computing farms, where thousands of central processing units (CPUs) are strung together to handle many jobs in parallel, have become ubiquitous to address the need for faster computation. However, CPU clock-speed gains began to decrease at about 3GHz, and CPUs turned to greater bit-widths and multi-core and multi-processor configurations so that their performance could continue to scale. Graphics processing unit (GPU)-acceleration offers the needed boost to CPU-only computing to address the on-going requirements of the semiconductor industry. Market-proven GPU-acceleration solutions combine the strengths of both CPU and GPU computing to achieve optimal acceleration and cost/performance ratios. The semiconductor-manufacturing environment is sensitive to any downtime, particularly in the 24/7 clean room. A computing platform with mean-time-between-failures and mean-time-to-repair good enough for the clean room is ready for any deployment in semiconductor manufacturing. Today, GPU-accelerated systems have been deployed successfully in semiconductor manufacturing by companies such as NuFlare and Advantest.

Introduction

Even with the innate limitations of 193i lithography, feature sizes are shrinking on the mask, while features on the wafer are shrinking even faster. Perhaps more importantly, the requirement for precision on both mask and wafer continues to increase. The long-awaited arrival of extreme ultraviolet (EUV) masks will mean that feature sizes will shrink dramatically and precision requirements will rise suddenly in one generation.

These trends put an unusual demand on semiconductor equipment manufacturers to provide more precision without impacting turnaround time. So, as the data sets upon which semiconductor design and manufacturing applications must operate grow ever larger and more complex, runtimes for processes at all stages of chip design and manufacturing have become a significant challenge for the semiconductor industry.

Over the years, several different approaches to massively parallel computing have been developed to address this issue, including Intel's MIC Architecture, IBM's Blue Gene, and coprocessors, such as digital-signal processors (DSPs) and field-programmable gate arrays (FPGAs). GPU-accelerated computing also emerged as an

attractive option for computation-intensive applications. At first, because of the consumer-market roots of GPUs, the scientific computing community was cautious in adopting GPU-accelerated applications for production use. However, today it is clear that GPU-accelerated computing already has begun to play a significant role in computing for semiconductor design and manufacturing.

The key to reaping the benefits of GPU-accelerated computing is to apply it where it will have the greatest impact, to balance it carefully with CPU-based computing, and to create robust, reliable systems around the GPUs that can perform even in the 24/7 operations of a clean room environment.

GPU-Acceleration: Gaming Roots, Expanding to Scientific Computing

GPUs were first developed as processing engines for the complex graphical content of computer games in the 1990s. This very lucrative and highly competitive market fueled the rapid growth and development of GPUs as sophisticated computing devices.

By the early 2000s, substantial clusters that included GPUs were being commissioned in research environments. But early generations didn't yet have the reliability and repeatability required for production or manufacturing applications. An incorrectly computed value in computer game graphics may result in a wrong pixel, which may not even be noticeable. However, in a semiconductor mask data processing program, the same incorrect value might endanger a multi-million-dollar investment in a chip.

Clearly, there was great potential for applying GPU-acceleration to scientific computing, and with the wave of innovations and evolution tracking along with Moore's Law, general-purpose graphics processing unit (GPGPU) computing became popular by the mid-2000s. GPGPUs became a separate product line for the GPU manufacturers, specifically addressing the needs of commercial users. The introduction of languages and frameworks such as Brook, CUDA, and OpenCL made GPGPUs a viable platform for application development. The leading GPU developers worked with the scientific computing community to bring proven, professional versions of their GPUs to market by the start of this decade. Floating point calculations of GPGPUs became IEEE compliant, just as those of CPUs. Repeatability for parallel computing is difficult, but the challenges are the same whether with CPU-only computing or with GPU-acceleration.

Today, robust and production-proven GPU-accelerated manufacturing equipment is deployed in risk-intolerant semiconductor manufacturing lines. For example, the Advantest E3640 MVM-SEM uses GPU-accelerated wafer plane analysis (WPA) of photomask images to predict how the image will look on the silicon wafer. The machine operator is able to decide in real-time whether those mask features are suitable for production. Another example is the NuFlare EBM-9500 eBeam-based mask writer, which uses GPU-accelerated, inline thermal-effect correction (TEC) to improve the quality of the photomask while significantly reducing the write time.

The low latency, precise calculations essential to both of these applications cannot be achieved with an off-the-shelf solution without GPU-acceleration.

Successful GPU-Acceleration is a Matter of Careful Balance

Adopting GPU-accelerated computing is not a simple matter of replacing CPUs with GPUs. A naive port of an application written for a CPU-based platform to a GPU-accelerated platform may only realize an acceleration of 3-4X. To realize the >10X acceleration potential of GPU-based computing, an application must be conceived with GPUs in mind and its algorithms designed to fully exploit the strengths – and avoid the weaknesses – of GPU-accelerated computing. The most important thing is to understand when to deploy GPUs and when to use CPUs. A robust GPU-acceleration approach enables a sophisticated software engineer to combine the strength of each to the benefit of the whole system.

While skillful engineering can often morph a given problem into one that is more suitable for one platform or the other, some generalizations hold true. GPUs excel at single-instruction, multiple data (SIMD) calculations, particularly when the application involves lots of things (like pixels or entries of a matrix) that require the same calculations, or where latency is critical, such as real-time or interactive computing involving simulation. Gaussian convolutions are one example of the kinds of computation at which GPUs excel.

Likewise, CPUs excel at heuristics that have many deductive branches, such as a series of if-then-else sequences that determine what to calculate depending on the situation.

Instead of just swapping out CPUs for GPUs, the real task is to combine some ratio of GPUs to CPUs in a single system. Clever engineering can optimize these combined resources for many computational problems, particularly scientific applications. The programming effort of a GPU-accelerated platform involves deciding what to put on CPUs, what to put on GPUs, then scheduling and load-balancing the two to achieve optimal overall performance.

The greatest overall acceleration gains result from loading GPUs only with computations of much higher advantage and using the otherwise idle cores of the CPUs that the GPU accompanies to handle other operations. In this way, every calculation within a system can be faster by assigning it to the optimal computing resource.

This balanced acceleration approach is applicable especially to real-time applications required for the in-line computations embedded in semiconductor manufacturing equipment. There are ample opportunities for software enhancement of such operations: lithography simulation, eBeam simulation, charging-effect correction, thermal-effect correction, process simulation, and practically any simulation of natural effects, which can benefit from GPU-acceleration.

The GPU "Sweet Spot"

The biggest difference between CPUs and GPUs is the number of computing threads available. A new-generation GPU can run 500 times more concurrent computing threads than a similar-generation CPU. This factor of 500 makes a big difference for certain types of computational tasks. Making any given computing sub-element run well on GPUs is about finding an approach to the problem that fully utilizes the 500X difference in the number of threads available.

This computing-thread advantage is why GPUs shine in situations where accuracy is critical but execution speed cannot be compromised. For instance, an engineer designing an application for a CPU-centric approach might need to abbreviate computing based on many specialized situations or contexts (as in simulations). In a CPU-centric environment, detailed, context-specific models are often the best way to balance accuracy with computation speed. However, context-specific models are inherently context-sensitive. These heuristics are always prone to loss of precision in the boundary between different contexts.

In contrast, an engineer designing for a GPU-based system might opt for a brute-force approach, where fundamental physical phenomena can be mathematically computed by taking advantage of the 500X difference in the available computing threads to provide the most accurate results. No context-specific models are required because every situation is computed accurately based on the underlying physics. Uniquely with GPU-acceleration, brute-force computation can be done in the same elapsed time as CPU-only computation using a platform of similar cost.

GPU-Computing Case Study: Cost/Performance Analysis

D2S first turned to GPU-acceleration in 2009 for its model-based mask data preparation (MB-MDP) technology. MB-MDP was developed in response to the increasing use of complex mask shapes – either Manhattan shapes with a large number of jogs, or curvilinear shapes such as ideal Inverse Lithography Technology (ILT) output – for masks targeting leading-edge semiconductor processes.

CPUs are designed for "if-then-else" algorithmic flexibility, so most traditional computational approaches to mask data preparation have been rule-based, where thousands of rules are used to define mask shapes and how they should be processed. When a complex shape doesn't exactly fit any of the rules in the database, traditional approaches "stitch" together an approximation of the complex shape using the rules at hand. However, this process is both time-consuming and error-prone. As masks became more and more complex, traditional CPU-only computation resulted in prohibitive processing times.

D2S recognized that Gaussian convolution was well-suited to GPU computation, and that a GPU-accelerated system could compute precisely an eBeam simulation (or other mask-effect simulations) for any arbitrary shape in about the same time as traditional rule-based approaches took to create an approximation of these shapes. Because an actual computation does not suffer from the stitching problems

often encountered in the traditional processing of complex shapes, the advantages of GPU-acceleration are particularly strong for complex masks.

As an illustration, D2S ran Gaussian convolution on an arbitrarily sized piece of mask data (~80µm by 80µm, 10nm pixels), using a node of its fourth-generation Computational Design Platform (CDP) to demonstrate runtimes for a CPU-only implementation, and for a CPU+GPU implementation. The D2S CDP node comprises two NVIDIA K-80 GPUs and two Intel Xeon E5-2630 v3 CPUs. The "CPU Only" implementation runs on one of the CPUs using all eight cores. The "CPU+GPU" implementation runs on one GPU and uses one core of one CPU (this GPU implementation mirrors the way the D2S CDP is used in actual semiconductor manufacturing applications – the other CPU cores are used to perform other operations). Algorithms and implementations were optimized for each platform separately. The runtimes are graphed in Figure 1 below. The CPU+GPU version is 10X faster.

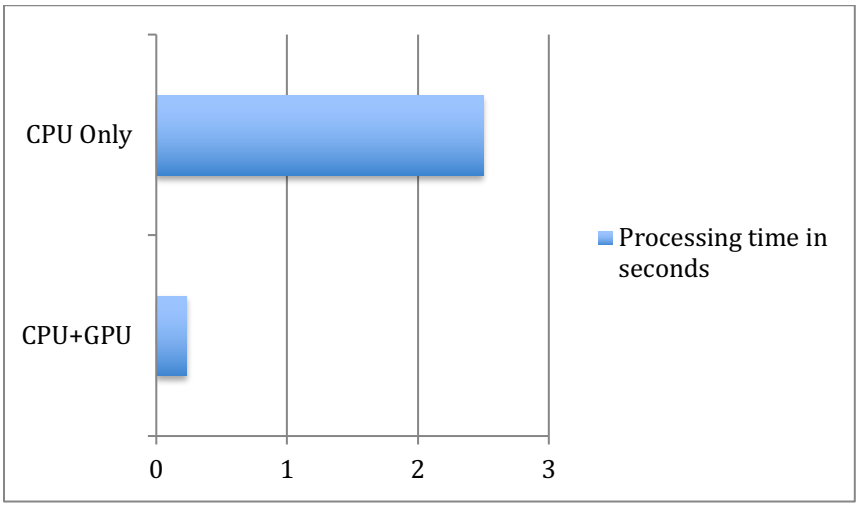

Figure 1. Gaussian convolution run on CPU+GPU and an eight-core CPU. (Data size: ~80 µm by 80µm, 10nm pixels).

Of course, economics play a part as well. A comparison of costs needs to take into account the costs of ownership, including power consumption and space requirements, which are especially important in a clean room. The D2S CDP uses two GPUs per board for heavy computing, and each GPU costs about $4200 and consumes 350W peak power. The two CPUs on each board are 2.6GHz (assuming Turbo speed when all cores are used) dual processors, which cost about $650 each and consume 85W peak power. The need to maintain a desirable thermal density prevents packing more CPUs or GPUs on the same board. To contrast, a CPU-only cluster would likely need to pack 20 dual processor boards into a 7U rackmount space, each board carrying a pair of E5-2699 v4 (22-core 2.8GHz assuming Turbo speed when majority of the cores are used) processors. This processor configuration costs about $4100 and consumes 145W peak power. CPUs offer many variations on clock speeds, numbers of cores and numbers of processors sharing the bus. Different applications optimize differently, and different choices of CPUs would offer

different trade-offs, but this is a good representative example of a "CPU-Only" configuration for data processing. Partitioning the problem, adding a halo region around each partition, and potentially communicating between partitions all add extra overhead, but we assume for this study that the overhead is zero. In reality, there would be less overhead for a "CPU+GPU" configuration because each partition can be larger.

Assuming a "CPU+GPU" configuration that is a D2S CDP node, and comparing that against the "CPU-Only" configuration described above, the comparison along various metrics would be as shown in Table 1 below. We compare in the same 7U rackmount space. The "CPU-Only" version would have 40 CPUs. The "CPU+GPU" version would have 14 CPUs and 14 GPUs.

In 7U Rackmount Space	**CPU-Only**	**CPU+GPU**
Processor Cost Per 7U	$164,000	$67,900
Power Consumed Per 7U	5800W	6090W
Speed of eBeam Simulation	2.27 units of time	1.67 units of time
Performance/$	2.7	8.8
Performance/Watt	76	98

Table 1. Cost/performance analysis for CPU only vs. CPU+GPU computing nodes.

The "CPU+GPU" combination has a substantially better performance per dollar with a slightly improved performance per watt.

Passing the Clean Room Reliability Test

There is nothing inherently different about computational algorithms that run in a semiconductor manufacturing clean room from any other environment where the system is run on a continuous, 24/7 basis. However, the computational system needs to meet the more stringent environmental requirements of the clean room, where even small variations in heat or contaminant levels are unacceptable. Also, there is a substantive difference in the design of the system that will meet the much more difficult and stringent requirements of installation and service in a clean room.

To service equipment in a clean room, all of the service personnel and equipment must be cleaned, and clothed or wrapped to meet clean-room standards, which can make even a simple repair unwieldy. The system must be designed to require a minimum of operational steps in the clean room. Second, any stoppage on the line in a clean room is very expensive, both for the manufacturer in terms of slowing throughput and for their customer, for whom a delay of a few days can mean millions of dollars in lost revenue. The system must be designed for high mean-time-between-failures and low mean-time-to-repair.

Any system designed for this challenging environment requires careful engineering for reliability, as well as for ease of service and recovery. All computing platforms, whether CPU-only or GPU-accelerated, are only as reliable as their most vulnerable component. But every system includes necessary components, such as power supplies, that will fail regularly under continuous use. It's not so much a question of "will something fail?", but rather "what happens when something does fail?" This is the real reliability challenge for deployment of any system used for semiconductor manufacturing.

There are two aspects to a robust and reliable GPU-accelerated system: redundancy and recovery. First, smart redundancy can help keep the system online. Hot-spares play an important part; for instance, when one power supply fails, a spare part already installed in the system is automatically called into service with no stoppage. Notifications enable technicians to replace the failed part without downtime for the system as a whole.

Recovery is the other important aspect of reliability. The complex computations performed in the semiconductor mask shop, for instance, may take 24 hours to run. If a node fails in hour 23, what happens? Must the operation start again from the beginning? Can the operation recover itself where it left off? Or, can the system move to hot spares and finish the operation on its own, if a bit slower (say, in 24 hours and 15 minutes)? And if a system reboot is necessary, does the reboot take days? Hours? Minutes? These are quite literally million-dollar questions.

The answers to these questions decide whether or not any computing platform is reliable enough to be used in this demanding environment. Today, GPU-accelerated systems are in deployment in multiple areas of semiconductor manufacturing, including in the clean-room environment. There is no question that GPU-accelerated computing is just as reliable and repeatable as CPU-only computing. Design for resilience, recoverability and serviceability make GPU-accelerated platforms appropriate for even the clean room.

Conclusion

In the never-ending quest for higher precision, better throughput, and more functionality demanded by the industry to keep up with Moore's Law, GPU-acceleration and related simulation-based technologies will play a key role. Because of its ability to shine in applications where accuracy is critical but execution speed cannot be compromised, it is clear that GPU-accelerated computing is the future for many applications in the world of semiconductor design and manufacturing with its unusual demand for ever-more precision without any change to turnaround time.

Successful GPU-accelerated systems are not a wholesale replacement of CPU-based computing, but rather a balance of GPUs with CPUs so that each processor is assigned the part of the task most suited to its architecture. Engineering skill is needed to build a GPU-computing platform that manages the balance of GPUs and

CPUs within the system. Field experience and expertise are required to create GPU-accelerated computing platforms that are robust and reliable enough for use in risk-averse semiconductor manufacturing facilities. With the right balance of computing resources, and with “clean room ready” reliability, GPU computing will play a significant role in the semiconductor manufacturing sector as the industry gears up to tackle its next set of challenges.