Whitepaper

NVIDIA GeForce GTX 980

Featuring Maxwell, The Most Advanced GPU Ever Made.

V1.0

Table of Contents

Introduction

In our 21-year quest to bring the most realistic 3D graphics to gamers, NVIDIA has introduced a number of innovations. With its hardware transform and lighting engine, NVIDIA's GeForce 256 ushered in the era of the GPU in 1999, bringing T&L support to consumer graphics for the first time. In late 2006 with the G80 GPU and the GeForce 8800 GTX graphics board, CUDA and the formalization of GPGPU computing as we know it today was first brought to the world. More recently, in 2010 we launched the GeForce GTX 480—our first GPU built using the Fermi graphics architecture. GeForce GTX 480 incorporated fifteen parallel tessellation units, enabling dramatic speedups in geometry processing compared to prior GPUs. With great strides in architectural efficiency, Kepler GPUs delivered significantly improved performance and power efficiency when introduced in 2012.

Earlier this year we launched the GeForce GTX 750 Ti, our first GPU from the Maxwell architectural family. Maxwell GPUs were designed from the ground up for extreme power efficiency and exceptional performance per watt consumed. In many DX11 applications, the GTX 750 Ti is capable of matching or even beating the performance of our once flagship GeForce GTX 480, while consuming only a fourth the power. Thanks to its remarkable power efficiency, our first generation Maxwell GPUs were ideal for use in power-limited environments like notebooks and small form factor PCs, in addition to mainstream desktops.

NVIDIA's latest GPU, GM204, is the first to use the full realization of our 10th generation GPU architecture, Maxwell. Our design goals for GM204 were to deliver:

- Extraordinary Gaming Performance for the Latest Displays
- Incredible Energy Efficiency
- Dramatic Leap Forward In Lighting With VXGI

Extraordinary Gaming Performance for the Latest Displays

After years of little or no progress in the evolution of PC displays, the pace of new technology introductions has dramatically increased. Screen resolutions are growing, with the first 4K displays for gamers hitting the market a little over a year ago. While 4K was initially expensive, thanks to rapidly falling prices, 4K adoption is now beginning to take off. In order to drive all these pixels, GM2xx GPUs were designed to deliver more performance at the highest resolutions, and also support new display tech for 4K like HDMI 2.0. In addition, with Dynamic Super Resolution, Maxwell introduces the ability to significantly enhance display quality on sub-4K displays by rendering at higher resolution and then applying an advanced resizing filter that leverages the high resolution information.

Virtual Reality displays are another area that is likely to drive a dramatic increase in rendering rate requirements. Two displays for two eyes doubles the rendering workload, and ever higher resolution and refresh rate capability will be crucial to eliminating rendering artifacts that break the illusion of reality for the VR user.

Incredible Energy Efficiency

All GeForce GPUs are made to deliver performance leadership in their respective segment. But PC gamers also expect efficiency—not noisy fans, excessive heat, and over-taxed power supplies. The desktop PC gaming market is seeing increasing popularity of smaller and thinner form factors, and the laptop PC gaming market has grown explosively in the past few years.

The Maxwell architecture was designed to provide an extraordinary leap in power efficiency and deliver unrivaled performance while simultaneously reducing power consumption from the previous generation. With a combination of advances originally developed for Tegra K1, new architectural approaches seen first in the GeForce GTX 750 Ti, and further refinement for GM204, Maxwell now delivers 2x the performance per watt compared to Kepler products.

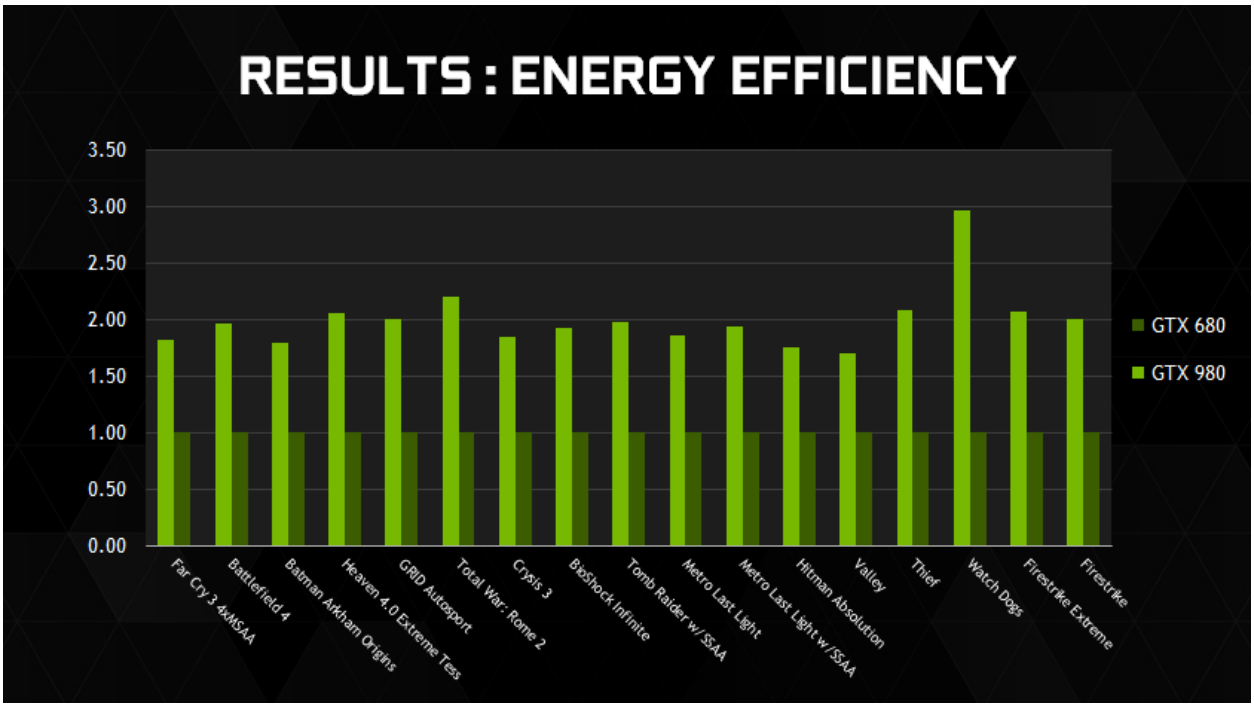


Figure 1: GM204 delivers 2X higher perf/watt compared to Kepler

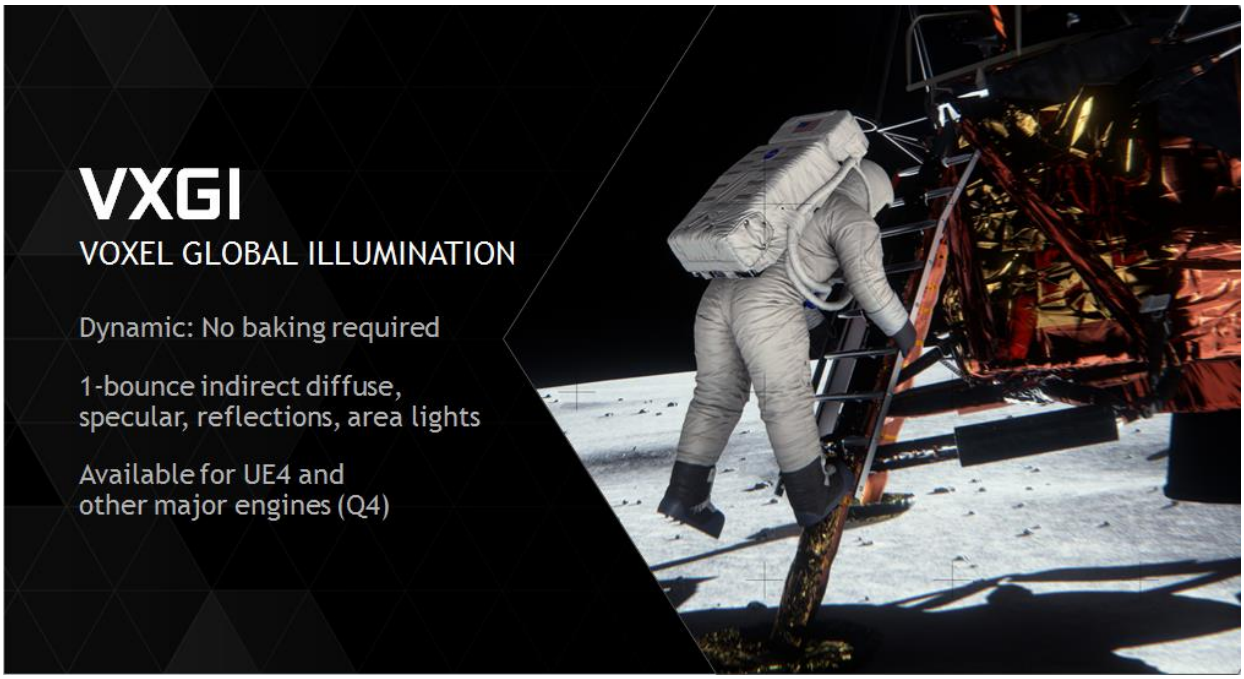
Dramatic Leap Forward In Lighting with VXGI

Lighting is the biggest hurdle in computer graphics that must be overcome in order to achieve photorealism. Global illumination gets us significantly closer to achieving this goal. Global illumination is often used in the movie industry to produce CG scenes that look stunning, but rendering truly dynamic global illumination in real time on the GPU has been impractical until now.

NVIDIA Voxel Global Illumination (VXGI) is the next big leap forward in lighting. VXGI is based on the concept pioneered by NVIDIA researcher Cyril Crassin in 2011, of using a 3D data structure ("voxels") to capture coverage and lighting information at every point in the scene. This data structure can then be traced during the final rendering stage to accurately determine the effect of light bouncing around in the scene.

Cyril's original implementation relied on voxels that were stored in an octree structure. While it was able to run successfully on a GeForce GTX 680, it had limitations. We've spent the last three years developing an implementation that can be accelerated natively by the GPU, as well as improving the algorithm. The result of that effort is VXGI.

VXGI provides dramatically improved performance for global illumination. While VXGI's software algorithm will run on all GPUs, the performance benefits of VXGI hardware acceleration will only be available on our latest Maxwell GPUs.



The first graphics cards to ship with our new GM204 GPU are the GeForce GTX 980 and GeForce GTX 970. For simplicity purposes we'll be focusing solely on the GeForce GTX 980 in this document. With 2048 CUDA Cores and 4GB of GDDR5 memory, GM204 is the fastest GPU in the world, yet with a TDP of just 165W, it's also the most efficient.

GM204 Hardware Architecture In-Depth

Figure 2: GM204 Full-chip block diagram

Like Fermi and Kepler, GM204 is composed of an array of Graphics Processing Clusters (GPCs), Streaming Multiprocessors (SMs), and memory controllers. GM204 consists of four GPCs, 16 Maxwell SMs (SMM), and four memory controllers. GeForce GTX 980 uses the full complement of these architectural components (if you are not well versed in these structures, we suggest you first read the Kepler and Fermi whitepapers).

Another version of the chip, with 13 SMs, will ship concurrently and be called GeForce GTX 970. In the future we plan to offer additional products based on GM204 that will ship with different combinations of GPCs, SMs, and memory controllers to address various segments of the graphics market.

In GeForce GTX 980, each GPC ships with a dedicated raster engine and four SMMs. Each SMM has 128 CUDA cores, a PolyMorph Engine, and eight texture units. With 16 SMMs, the GeForce GTX 980 ships with a total of 2048 CUDA cores and 128 texture units.

The GeForce GTX 980 features four 64-bit memory controllers (256-bit total). Tied to each memory controller are 16 ROP units and 512KB of L2 cache. The full chip ships with a total of 64 ROPs and 2048KB of L2 cache (this compared to 32 ROPs and 512K L2 on GK104).

The following table provides a high-level comparison of Maxwell vs. our previous-generation GK104 GPU:

GPU	GeForce GTX 680 (Kepler)	GeForce GTX 980 (Maxwell)
SMs	8	16
CUDA Cores	1536	2048
Base Clock	1006 MHz	1126 MHz
GPU Boost Clock	1058 MHz	1216 MHz
GFLOPs	3090	4612[1]
Texture Units	128	128
Texel fill-rate	128.8 Gigatexels/sec	144.1 Gigatexels/sec
Memory Clock	6000 MHz	7000 MHz
Memory Bandwidth	192 GB/sec	224 GB/sec
ROPs	32	64
L2 Cache Size	512KB	2048KB
TDP	195 Watts	165 Watts
Transistors	3.54 billion	5.2 billion
Die Size	294 mm²	398 mm²
Manufacturing Process	28-nm	28-nm

The GeForce GTX 980 has double the SMs compared to the GK104 GPU used in the GeForce GTX 680 released two years ago. Because of the changes implemented in GTX 980's new Maxwell SM, we were able to integrate 2x more SMs without doubling the die size. With each SM also containing its own dedicated PolyMorph Engine, GeForce GTX 980 also has twice the number of geometry units as its direct predecessor. We'll be discussing more details on the new SM design in the next section of the whitepaper.

Based on efficiency and workload analysis, and math vs. texture processing requirements of modern games, NVIDIA engineers determined that eight texture units per SMM is the best architectural balance for Maxwell; therefore, the total number of texture units is the same as Kepler, 128. However, thanks to GeForce GTX 980's higher clocks, texture fill rate improves by 12% from one generation to the next. To improve performance in high AA/high resolution gaming scenarios, we doubled the number of ROPs

[1] *The GFLOPS and texel fill rates in this chart are based on GPU Base Clock*

from 32 to 64. Again, thanks to the added benefit of higher clocks, pixel fill-rate is actually more than double that of GTX 680: 72 Gpixels/sec for GTX 980 versus 32.2 Gpixels/sec for GTX 680.

The memory subsystem has also been significantly revamped. GTX 980's memory clock is over 15% higher than GTX 680, and GM204's cache is larger and more efficient than Kepler's design, reducing the number of memory requests that have to be made to DRAM. Improvements in our implementation of memory compression provide a further benefit in reducing DRAM traffic—effectively amplifying the raw DRAM bandwidth in the system.

Maxwell Streaming Multiprocessor

Figure 3: GM204 SMM Diagram (GM204 also features 4 DP units per SMM, which are not depicted on this diagram)

The SM is the heart of our GPUs. Almost every operation flows through the SM at some point in the rendering pipeline. Maxwell GPUs feature a new SM that's been designed to provide dramatically improved performance per watt than prior GeForce GPUs.

Compared to GPUs based on our Kepler architecture, Maxwell's new SMM design has been reconfigured to improve efficiency. Each SMM contains four warp schedulers, and each warp scheduler is capable of dispatching two instructions per warp every clock. Compared to Kepler's scheduling logic, we've integrated a number of improvements in the scheduler to further reduce redundant re-computation of scheduling decisions, improving energy efficiency. We've also integrated a completely new datapath organization. Whereas Kepler's SM shipped with 192 CUDA Cores—a non-power-of-two organization—the Maxwell SMM is partitioned into four distinct 32-CUDA core processing blocks (128 CUDA cores total per SM), each with its own dedicated resources for scheduling and instruction buffering. This new configuration in Maxwell aligns with warp size, making it easier to utilize efficiently and saving area

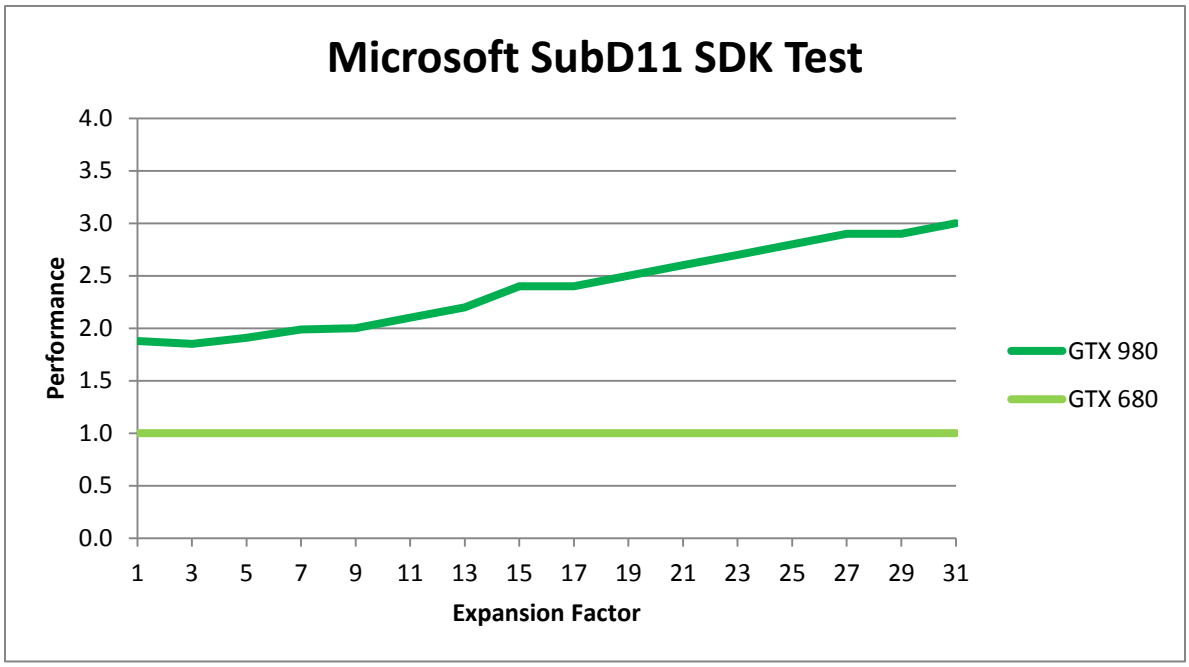
and power that had to be spent to manage data transfer in the more complex datapath organization used by Kepler.

Compared to Kepler, the SMM's memory hierarchy has also changed. Rather than implementing a combined shared memory/L1 cache block as in Kepler SMX, Maxwell SMM units feature a 96KB dedicated shared memory, while the L1 caching function has been moved to be shared with the texture caching function.

As a result of these changes, each Maxwell CUDA core is able to deliver roughly 1.4x more performance per core compared to a Kepler CUDA core, and 2x the performance per watt. At the SM level, with 33% fewer total cores per SM, but 1.4x performance per core, each Maxwell SMM can deliver total per-SM performance similar to Kepler's SMX, and the area savings from this more efficient architecture enabled us to then double up the total SM count, compared to GK104.

PolyMorph Engine 3.0

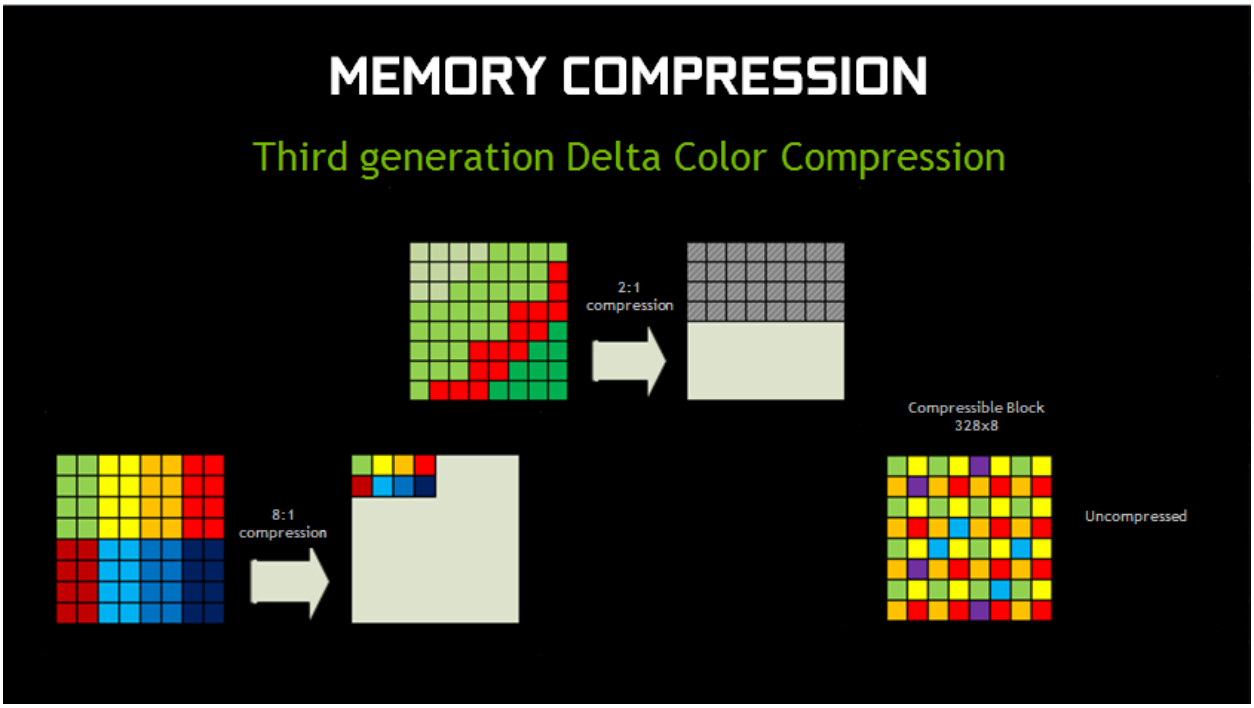
Tessellation was one of DirectX 11's key features and will play a bigger role in the future as the next generation of games are designed to use more tessellation. With the addition of more SMs in GM204, GTX 980 also benefits from 2x the Polymorph Engines, compared to GTX 680. As a result, performance on geometry heavy workloads is roughly doubled, and due to architectural improvements within the PE, can achieve up to 3x performance improvement with high tessellation expansion factors.



GM204 Memory Subsystem

In GM204 one ROP partition contains 16 ROP units (compared to eight ROP units per partition in Kepler); each ROP can process a single color sample. With four ROP partitions, a full GM204 has 64 ROPs, twice that of its predecessor, GK104, dramatically improving ROP throughput.

GM204 has a 256-bit memory interface with 7Gbps GDDR5 memory, the fastest in the industry. GM204 also features a unified 2048KB L2 cache that is shared across the GPU. In addition, GM204 has made significant enhancements to our memory compression implementation.



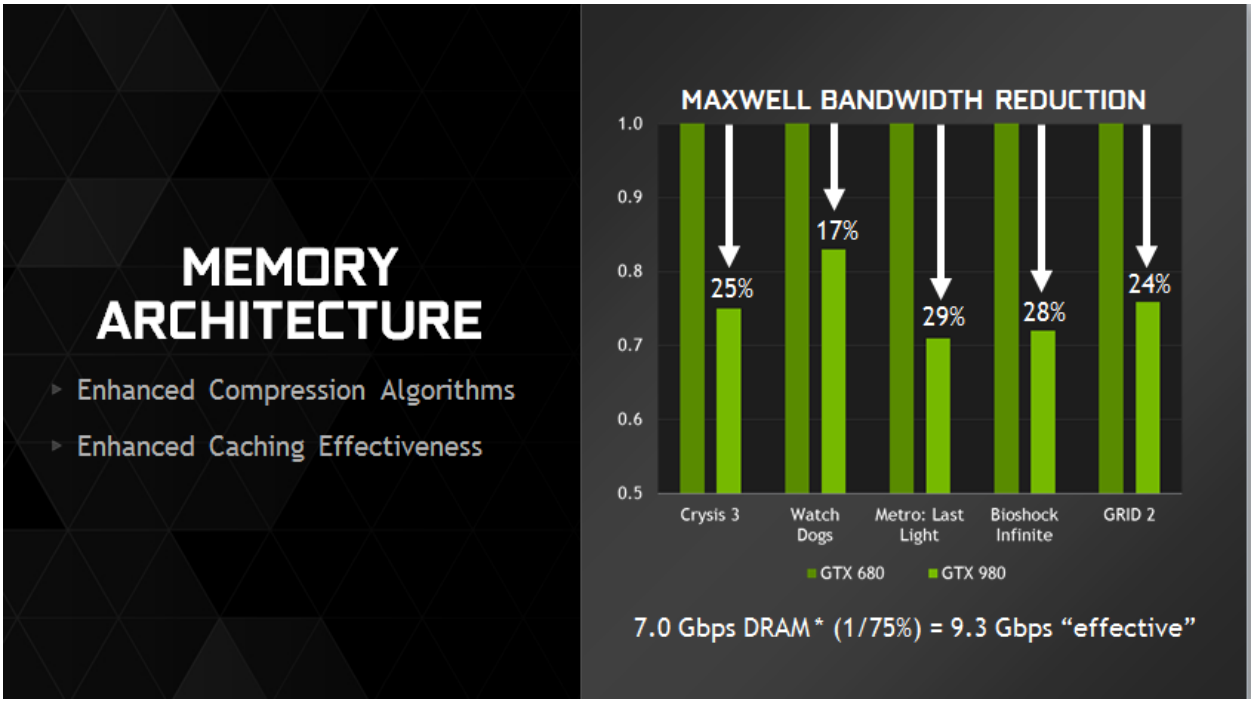
To reduce DRAM bandwidth demands, NVIDIA GPUs make use of lossless compression techniques as data is written out to memory. The bandwidth savings from this compression is realized a second time when clients such as the Texture Unit later read the data. As illustrated in the preceding figure, our compression engine has multiple layers of compression algorithms. Any block going out to memory will first be examined to see if 4x2 pixel regions within the block are constant, in which case the data will be compressed 8:1 (i.e., from 256B to 32B of data, for 32b color). If that fails, but 2x2 pixel regions are constant, we will compress the data 4:1.

These modes are very effective for AA surfaces, but less so for 1xAA rendering. Therefore, starting in Fermi we also implemented support for a “delta color compression” mode. In this mode, we calculate the difference between each pixel in the block and its neighbor, and then try to pack these different values together using the minimum number of bits. For example if pixel A’s red value is 253 (8 bits) and pixel B’s red value is 250 (also 8 bits), the difference is 3, which can be represented in only 2 bits.

Finally, if the block cannot be compressed in any of these modes, then the GPU will write out data uncompressed, preserving the lossless rendering requirement.

However, the effectiveness of delta color compression depends on the specifics of which pixel ordering is chosen for the delta color calculation. Maxwell contains our third generation of delta color compression, which improves effectiveness by offering more choices of delta calculation to the compressor.

Thanks to the improvements in caching and compression in Maxwell, the GPU is able to significantly reduce the number of bytes that have to be fetched from memory per frame. In tests with a variety of games, Maxwell uses roughly 25% fewer bytes per frame compared to Kepler. This means that from the perspective of the GPU core, a Kepler-style memory system running at 9.3Gbps would provide effective bandwidth similar to the bandwidth that Maxwell's enhanced memory system provides.

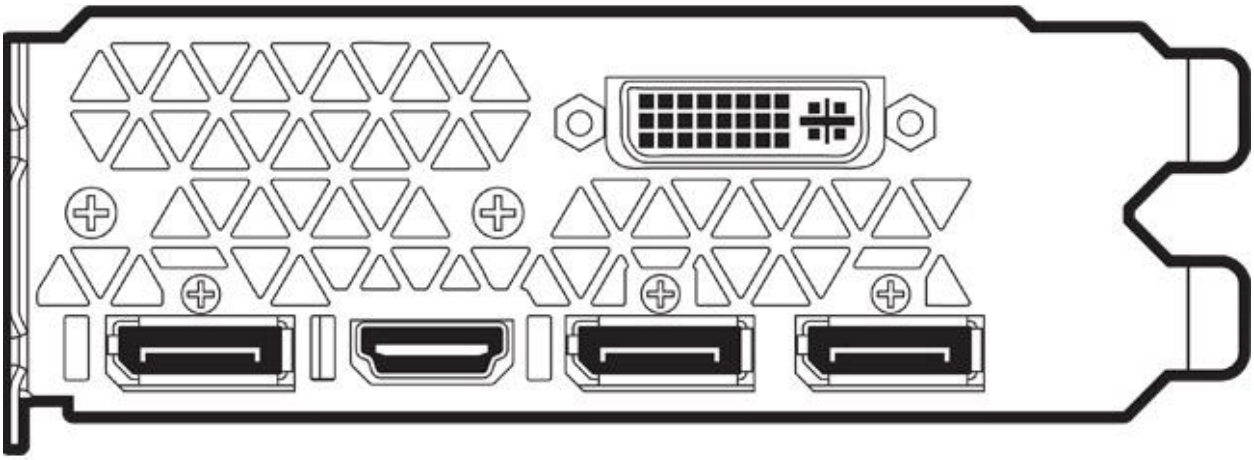


New Display and Video Engines

As the rapid adoption rate of 4K displays shows, consumer demand for high resolution devices has intensified this year. To meet this demand, all Maxwell GPUs ship with a display engine capable of supporting resolutions up to 5K. In addition, up to four 4K MST displays can be driven by the GPU, compared to a maximum of two 4K MST displays in Kepler.

GeForce GTX 980 is also the world's first GPU to support HDMI 2.0. Whereas the previous generation, HDMI 1.4, can support 4K display at 30Hz for "444" RGB pixels, and 60Hz for "420" YUV pixels, with HDMI 2.0, the GPU can now drive full-resolution "444" RGB pixels at 4K resolution at 60Hz.

The GeForce GTX 980 reference board design ships with three DisplayPort 1.2 connectors, one HDMI 2.0 connector, and one dual-link DVI connector. Up to four display heads can be driven simultaneously from one card.



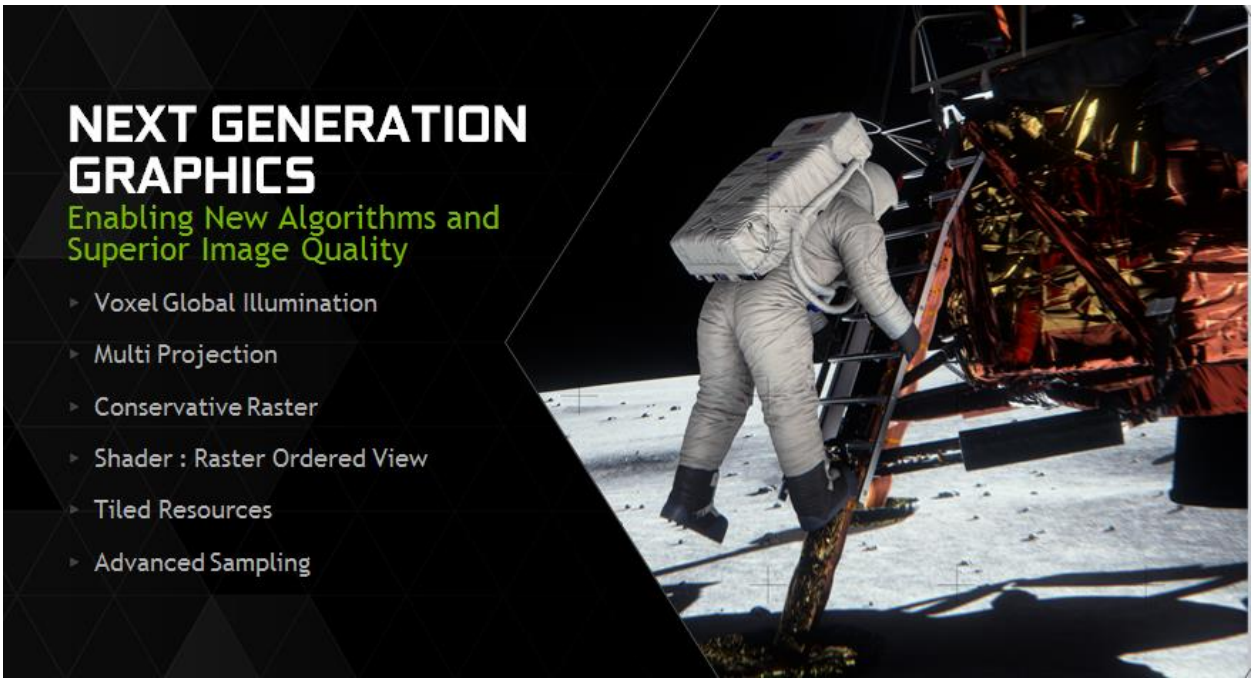
	Kepler	Maxwell
Number of Heads	4	4
Number of Links	6	7
Max Pixel Clock	540 MHz	1045 MHz
Max Resolution	3840 x 2160 @ 60Hz	5120 x 3200 @ 60Hz
Digital Protocols	LVDS, TMDS/HDMI 1.4, DP 1.2	LVDS, TMDS/HDMI 2.0, DP 1.2, eDP 1.4

When combined with a G-SYNC display, the GeForce GTX 980 delivers a gaming experience that's free of the distracting screen tearing that currently plagues gaming when Vsync is disabled. G-SYNC also eliminates display-subsystem-generated stutter and reduces input lag that gamers put up with today. Utilizing DisplayPort, the GeForce GTX 980 can drive up to three G-SYNC displays in Surround.

GM2xx Maxwell also ships with an enhanced NVENC encoder that adds support for H.265 (also known has HEVC) encoding. H.265 compression offers significant bandwidth savings versus H.264 at the same quality level, which will help enable the next generation of streamed gaming experiences within the home or powered by the cloud.

In addition, Maxwell's video encoder improves H.264 video encode throughput by 2.5x over Kepler, enabling it to encode 4K video at 60 fps. We're supporting this feature today with ShadowPlay.

Maxwell: Enabling The Next Frontier in PC Graphics



Correct modelling of lighting is the most computationally difficult problem in graphics, and with Maxwell, our objective was to make an enormous leap forward in the capability of the GPU to perform near-photo-real lighting calculations in real time.

In the real world, all objects are lit by a combination of direct light (photons that travel directly from a light source to illuminate an object) and indirect light (photons that travel from the light source, hit one object and bounce off of it and then hit a second object, thus indirectly illuminating that object). “Global illumination” (GI) is a term for lighting systems that model this effect. Without indirect lighting, scenes can look harsh and artificial. However, while light received directly is fairly simple to compute, indirect lighting computations are highly complex and computationally heavy.

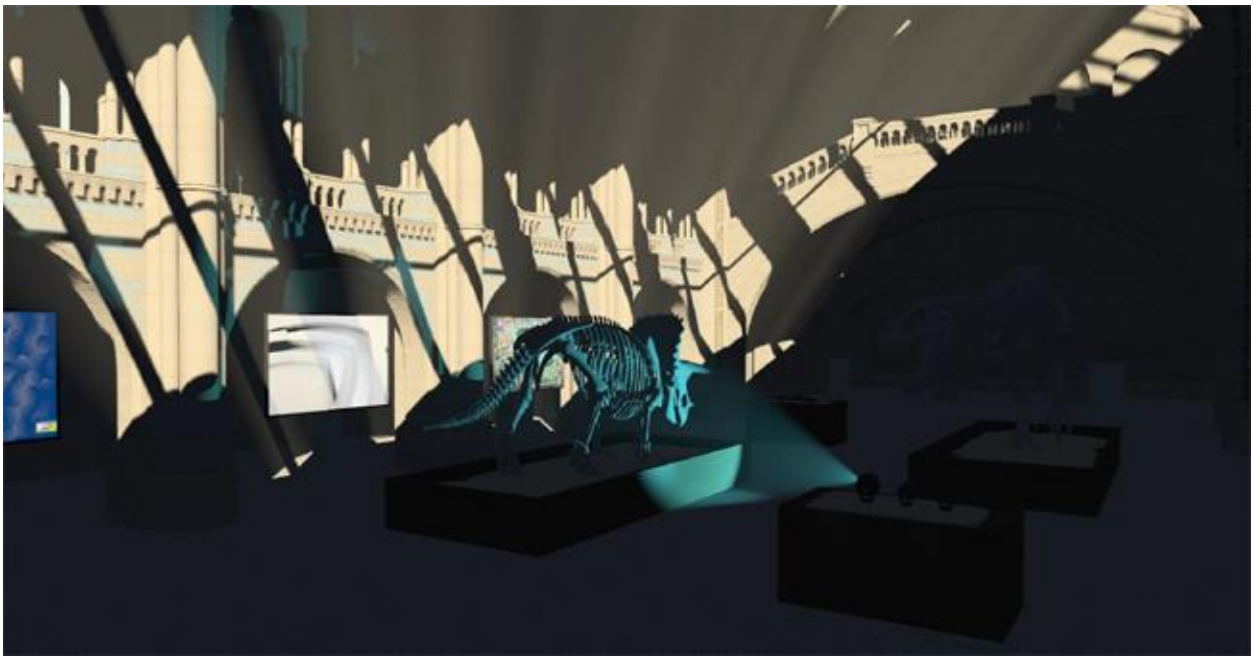


Figure 4: Scene rendered with direct lighting only

Figure 5: Same scene with GI enabled, note the indirect light and specular reflections on the floor

Because it's a computationally expensive lighting technique (particularly in highly detailed scenes), GI has been primarily used to render complex CG scenes in movies using offline GPU rendering farms. While some forms of GI have been used in many of today's most popular games, their implementations have relied on pre-computed lighting. These "prebaked" techniques are used for performance reasons; however, they require additional artwork, as the desired lighting effects must be computed beforehand.

Because prebaked lighting is not dynamic, it's often difficult or impossible to update the indirect light sources when in-game changes occur; say for instance an additional light source is added or something in the scene moves or is destroyed. Prebaked indirect lighting models the static objects of the scene, but doesn't properly apply to the animated characters or moving objects.

In 2011, NVIDIA engineers developed and demonstrated an innovative new approach to computing a fast, approximate form of global illumination dynamically in real time on the GPU. This new GI technology uses a voxel grid to store scene and lighting information, and a novel voxel cone tracing process to gather indirect lighting from the voxel grid. NVIDIA's Cyril Crassin describes the technique in his paper on the topic and a video from GTC 2012 is available here. Epic's *'Elemental'* Unreal Engine 4 tech demo from 2012 used a similar technique.

Figure 6: Epic's UE4 'Elemental' tech demo used voxel cone tracing for its jaw-dropping GI

Since that time, NVIDIA has been working on the next generation of this technology—**VXGI**—that combines new software algorithms and special hardware acceleration in the Maxwell architecture.

To understand how voxel global illumination works, it is helpful to first understand voxels. The term "voxel" is related to "pixel." Whereas a pixel represents a 2D point in space, a voxel represents a small cube (a volume) of 3D space. To perform global illumination, we need to understand the light emitting from all of the objects in the scene, not just the direct lights. To accomplish this, we dice the entire 3D space of the scene in all three dimensions, into small cubes called voxels. "Voxelization" is the process of determining the content of the scene at every voxel, analogous to "rasterization" which is the process of determining the value of a scene at a given 2D coordinate.

The following picture illustrates this approach applied to a single triangle. In the picture, each cube that contains a portion of the purple triangle is highlighted in blue.

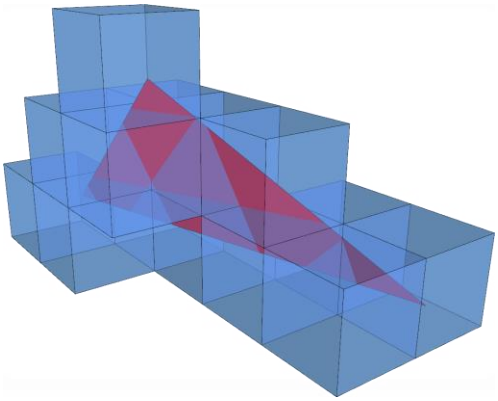


Figure 7: Voxels touched by a triangle

In VXGI, we store two pieces of key information in each voxel: (a) the fraction of the voxel that contains an actual object, and (b) for any voxel that contains an object, the properties of light coming from that object (i.e., bouncing off of it from primary light sources), including direction and intensity.

The first step, as illustrated in the following figure, is the coverage calculation step. In this step, each triangle needs to be checked from the perspective of each face of the cube, to assess what fraction of the voxel is covered. The picture on the left shows a traditional rasterized image of a simple scene. The picture on the right is a visualization of the voxelized result. In this picture, empty voxels are not drawn, voxels that are fully covered are drawn in red, and voxels that are partially covered are represented by a shade between blue (minimally covered) and red (fully covered). Note that the interior of each box is solid red, as expected, while the edges are blue, due to the edge of the triangle intersecting but not fully covering a voxel.

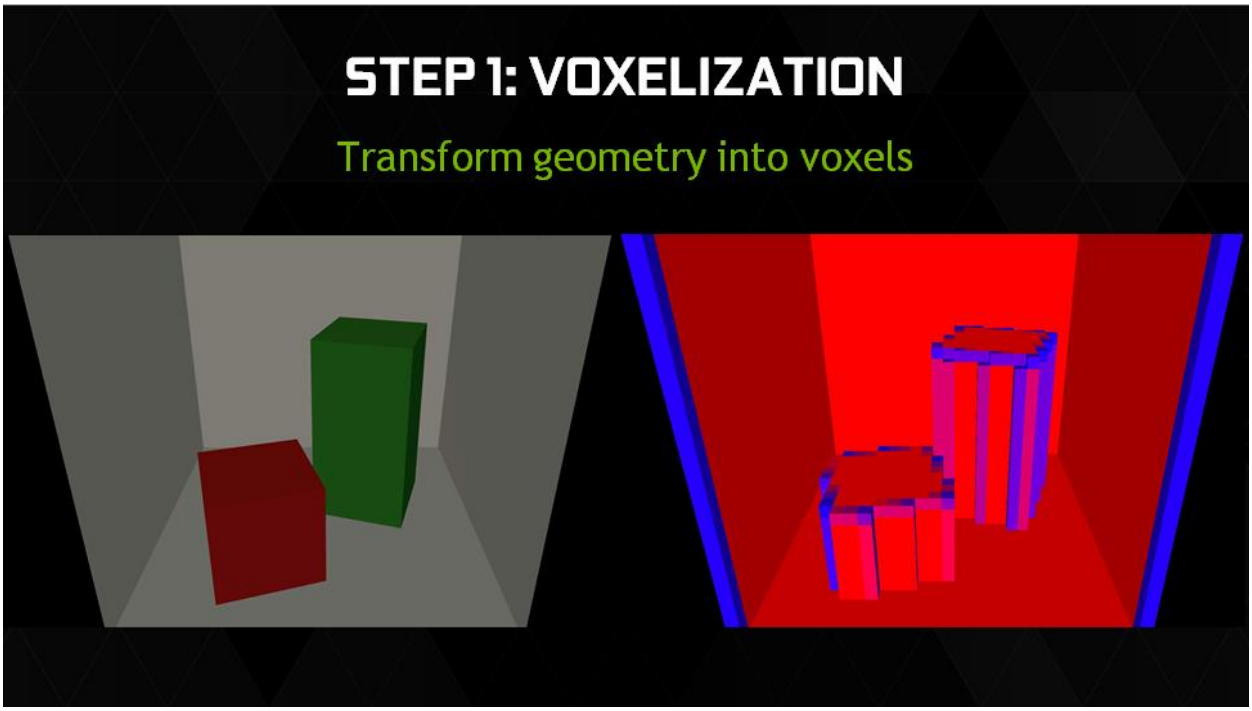


Figure 8: Voxelization results

Once the voxel coverage stage is complete, we store information into each voxel describing how the physical geometry will respond to light. This includes encoding the material's opacity and emissive/reflective properties. We then evaluate direct lighting at each non-empty voxel, and render the scene multiple times from the point of view of different light sources, capturing the amount of light that hits each voxel.

In the figure below, the direct light source indicated by the yellow dot causes light to strike the white walls and some of the surfaces of the red and green boxes. Each will then emit reflected light based on the color of the object (i.e., the white wall will emit a small amount of reflected white light, while the green box will emit reflected green light).

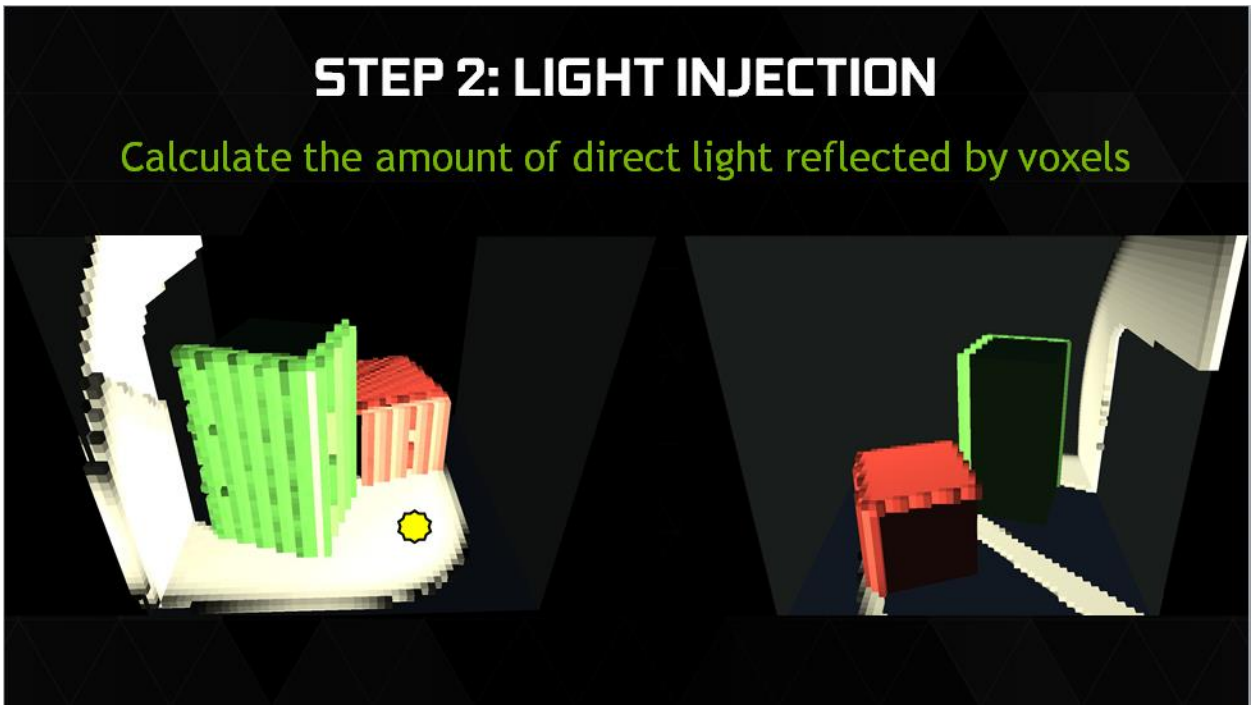
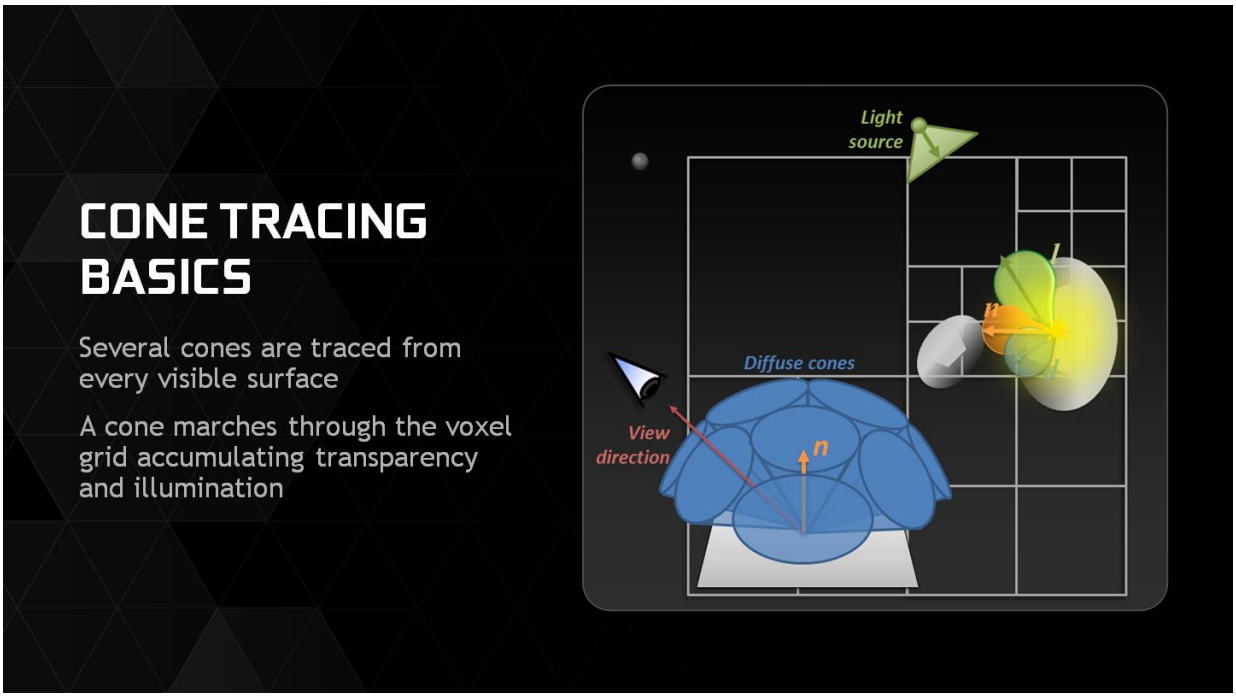


Figure 9: Results of light injection

After these steps, we now have a complete voxel data structure representing the lighting information in the scene.

The next and final step is to rasterize the scene. This step is largely the same as it would be for a scene rendered with other lighting approaches; the main difference is that the final rasterization and lighting now has a new and more powerful data structure (the voxel data structure) that it can use in its lighting calculations, along with other structures such as shadow maps.

The approach of calculating indirect lighting during the final rendering pass of VXGI is called cone tracing. Cone tracing is an approximation of the effect of secondary rays that are used in ray tracing methods. Using cones results in very realistic approximations of global illumination at a much lower computational cost than standard ray tracing.



Rendering a real-time reflection from a glossy curved surface has been difficult in the past. In order to render glossy reflections, you would traditionally need to launch hundreds or thousands of scattered secondary rays for each ray that bounces from the original reflector. It's incredibly challenging to reflect these lights realistically, especially when you also factor in the material properties of the various light reflectors. Using our approach, we've replaced the thousands of secondary rays with just a handful of voxel cones that are traced through the voxel grid.

We also use the same approach to quickly compute diffuse or specular lighting with only a few scattered cones. Ultimately as a result, we're able to compute approximate GI at high frame rates in real time, allowing us to realistically render glossy and metallic surfaces.

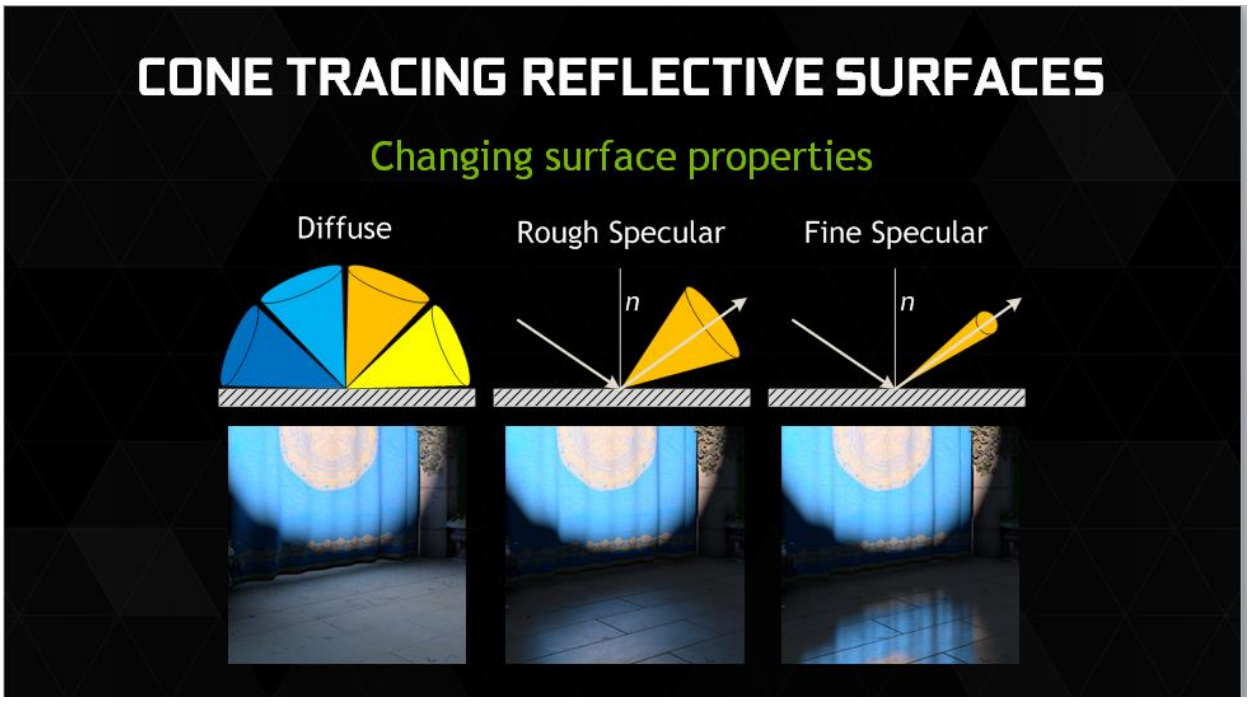


Figure 10: In the example above voxel cones are used to produce various forms of diffuse and specular light

Hardware Acceleration for VXGI – Multi-Projection and Conservative Raster

One exciting property of VXGI is that it is very scalable—by changing the density of the voxel grid, and the amount of tracing of that voxel grid that is performed per pixel, it is possible for VXGI to run across a wide range of hardware, including Kepler GPUs, console hardware, etc. However, for Maxwell it was an important goal to identify opportunities for significant acceleration of VXGI that would enable us to demonstrate its full potential and achieve the highest possible level of realism.

As described above, VXGI based lighting has three major phases—the first two are new, accomplishing the generation of a new voxel data structure, while the third stage is a modification of the existing lighting phase of real time rendering. Therefore, to enable VXGI as a real-time dynamic lighting technique, it is important that the new work—the creation of the voxel data structure—is as fast as possible, as this is the part of VXGI that is new work for the renderer. Fast voxelization ensures that changes in lighting or the position of objects in the scene can be reflected immediately in the lighting calculation.

With this in mind, it was a top priority for Maxwell to implement hardware acceleration for this stage. One important observation is that the voxelization stage is challenged by the need to analyze the same scene geometry from many views—each face of the voxel cube—to determine coverage and lighting. We call this property of rendering the same scene from multiple views "multi-projection." It turns out that multi-projection is a property of other important rendering algorithms as well. For example, cube maps (used commonly for assisting with modelling of reflections) require rendering to six faces. And as will be discussed in more depth later, shadow maps can also be rendered at multiple resolutions. Therefore, acceleration of multi-projection is a broadly useful capability. Today, multi-projection can be implemented either by explicitly sending geometry to the hardware multiple times, or by expanding geometry in the geometry shader; however, neither approach is particularly efficient. The specific capability that we added to speed up multi-projection is called "**Viewport Multicast**." With this feature, Maxwell can use dedicated hardware to automatically broadcast input geometry to any number of desired render targets, avoiding geometry shader overhead. In addition, we added some hardware support for certain kinds of per viewport processing that are important to this application.

"**Conservative Raster**" is the second feature in Maxwell that accelerates the voxelization process. As illustrated in the following *Figure 11*, conservative raster is an alternate algorithm for triangle rasterization. In traditional rasterization, a triangle covers a pixel if it covers a specific sample point within that pixel, for example, the pixel center in the following picture. Therefore with traditional rasterization, the four purple pixels would be considered "covered" by the triangle.

With conservative rasterization rules on the other hand, a pixel is considered covered if **any** part of the pixel is covered by any part of the triangle. In the following picture, the seven orange pixels are also "covered" by conservative rasterization rules.

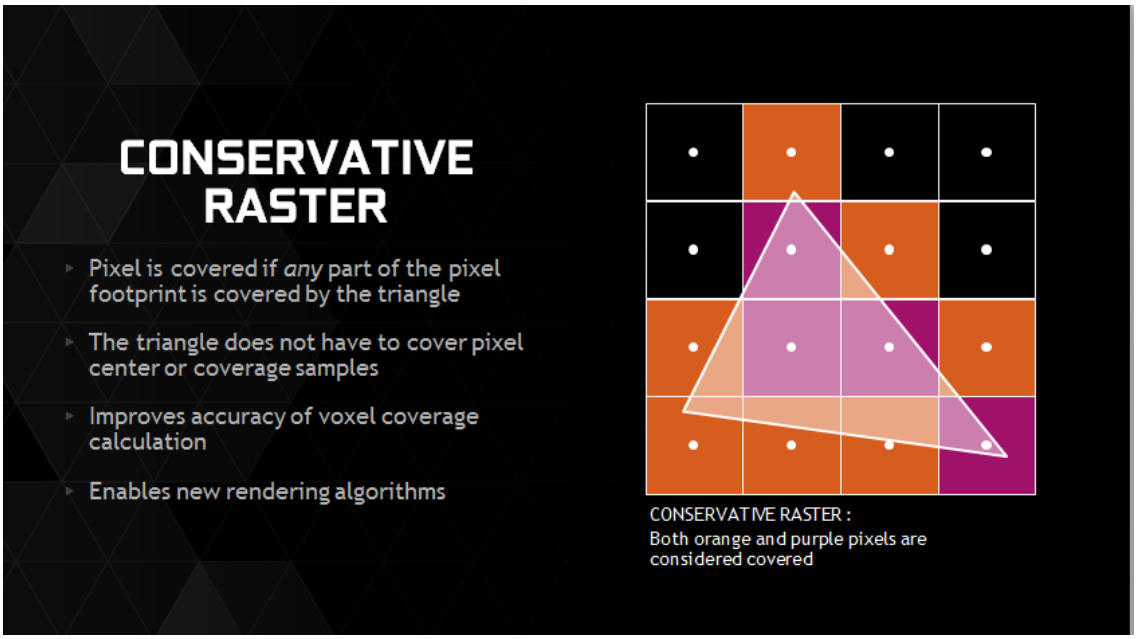


Figure 11: Illustration of conservative raster coverage rules

Hardware support for conservative raster is very helpful for the coverage phase of voxelization. In this phase, fractional coverage of each voxel needs to be determined with high accuracy to ensure the voxelized 3D grid represents the original 3D triangle data properly. Conservative raster helps the hardware to perform this calculation efficiently; without conservative raster there are workarounds that can be used to achieve the same result, but they are much more expensive.

The benefit of these features can be measured by running the voxelization stage of VXGI both ways (i.e., with the new features enabled vs. disabled). *Figure 12* below compares the performance of voxelization on "San Miguel," a popular test scene for global illumination algorithms—GTX 980 achieves a 3x speedup when these features are enabled.

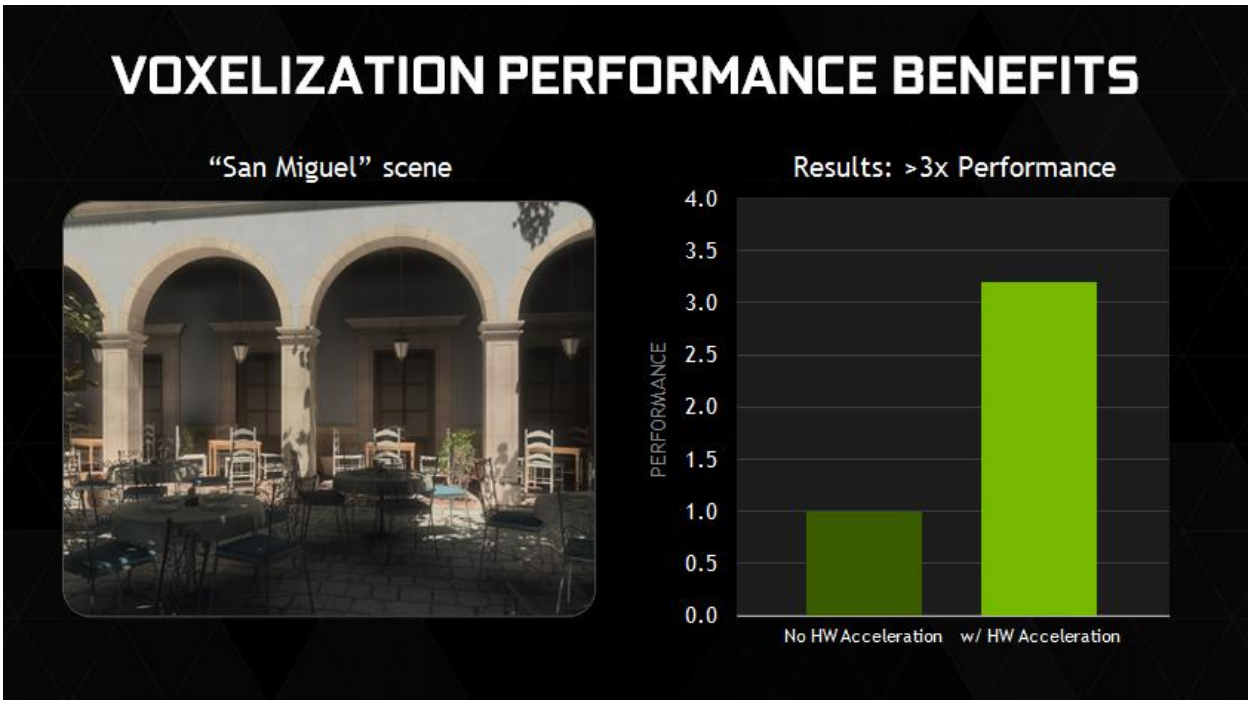


Figure 12: Voxelization performance with Maxwell hardware acceleration

Tiled Resources

DirectX 11.2 introduced a feature called Tiled Resources that could be accelerated with an NVIDIA Kepler and Maxwell hardware feature called **Sparse Texture**. With Tiled Resources, only the portions of the textures required for rendering are stored in the GPU's memory. Tiled Resources works by breaking textures down into tiles (pages), and the application determines which tiles might be needed and loads them into video memory. It is also possible to use the same texture tile in multiple textures without any additional texture memory cost; this is referred to as aliasing. In the implementation of voxel grids, aliasing can be used to avoid redundant storage of voxel data, saving significant amounts of memory. You can read more about Tiled Resources at this link.

One interesting application of Tiled Resources is multi resolution shadow maps. In the following *Figure 13*, the image on the left shows the result of determining shadow information from a fixed resolution shadow map. In the foreground, the shadow map resolution is not adequate, and blocky artifacts are clearly visible. One solution would be to use a much higher resolution shadow map for the whole scene, but this would be expensive in memory footprint and rendering time. Alternatively, with Tiled Resources it is possible to render multiple copies of the shadow map at different resolutions, each populated only where that level of resolution detail is needed based on the scene. In the image on the right, each resolution of shadow map is illustrated with a different color. The highest resolution shadow map (in red) is only used in the foreground when that high resolution is required.

Figure 13: Fixed resolution vs multi resolution shadow map quality

This is another application of multi-projection that will benefit from the hardware acceleration in Maxwell. In the future, we also believe that tiled resources can be leveraged within VXGI, to save voxel memory footprint.

Raster Ordered View

To ensure that rendering results are predictable, the DX API has always specified "in order" processing rules for the raster pipeline, in particular the Color and Z units ("ROP"). Given two triangles sent to the GPU in order—first triangle "A," then "B"—that touch the same XY screen location, the GPU hardware guarantees that triangle "A" will blend its color result before "B" blends it. Special interlock hardware in the ROP is responsible for enforcing this ordering requirement.

DX11 introduced the capability for the pixel shader to bind "Unordered Access Views" of color and Z buffers, and read and write arbitrary locations within those buffers. However as the name implies, there is no processing order guarantee when multiple pixel shaders are accessing the same UAV.

The next generation DX API introduces the concept of a "**Raster Ordered View**," which supports the same guaranteed processing order that has traditionally been supported by Z and Color ROP units. Specifically, given two shaders A and B, each associated with the same raster X and Y, hardware must guarantee that shader A completes all of its accesses to the ROV before shader B makes an access.

To support Raster Ordered View, Maxwell adds a new interlock unit in the shader with similar functionality to the unit in ROP. When shaders run with access to a ROV enabled, the interlock unit is

responsible for tracking the XY of all active pixel shaders and blocking conflicting shaders from running simultaneously.

One potential application for Raster Ordered View is order independent transparency rendering algorithms, which handle the case of an application that is unable to pre-sort its transparent geometry by instead having the pixel shader maintain a sorted list of transparent fragments per pixel.

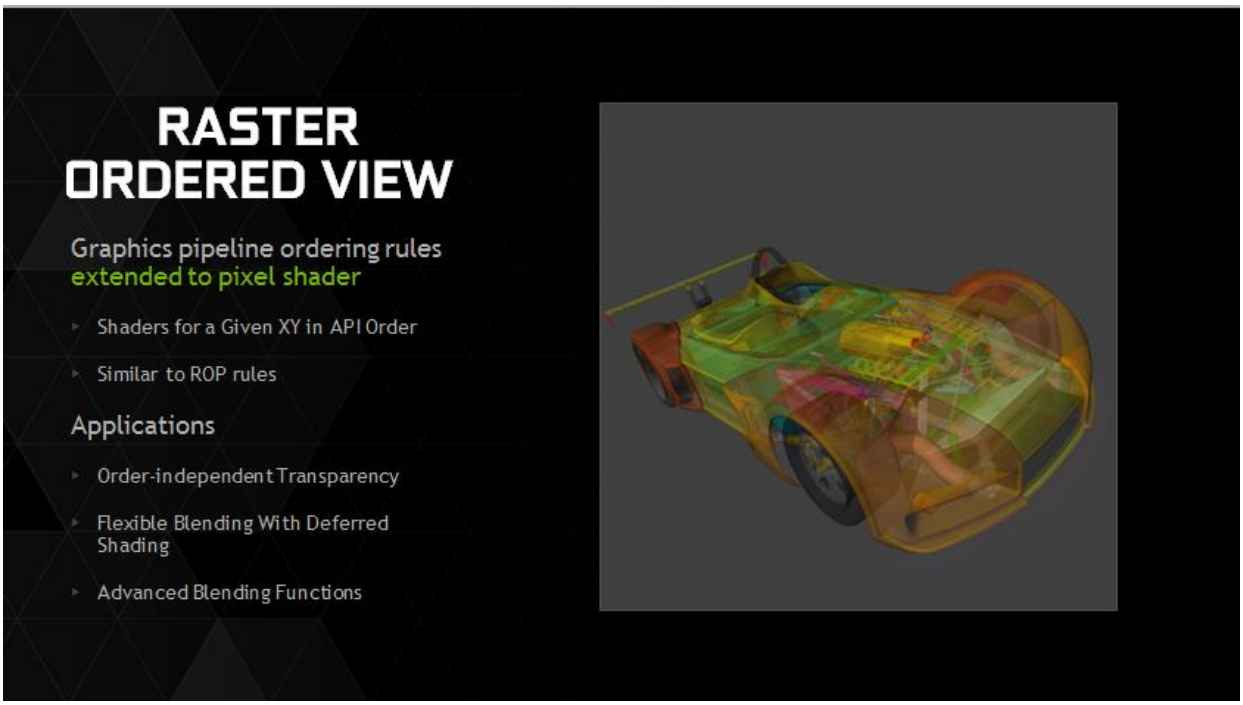


Figure 14: Raster Ordered View

DirectX 12

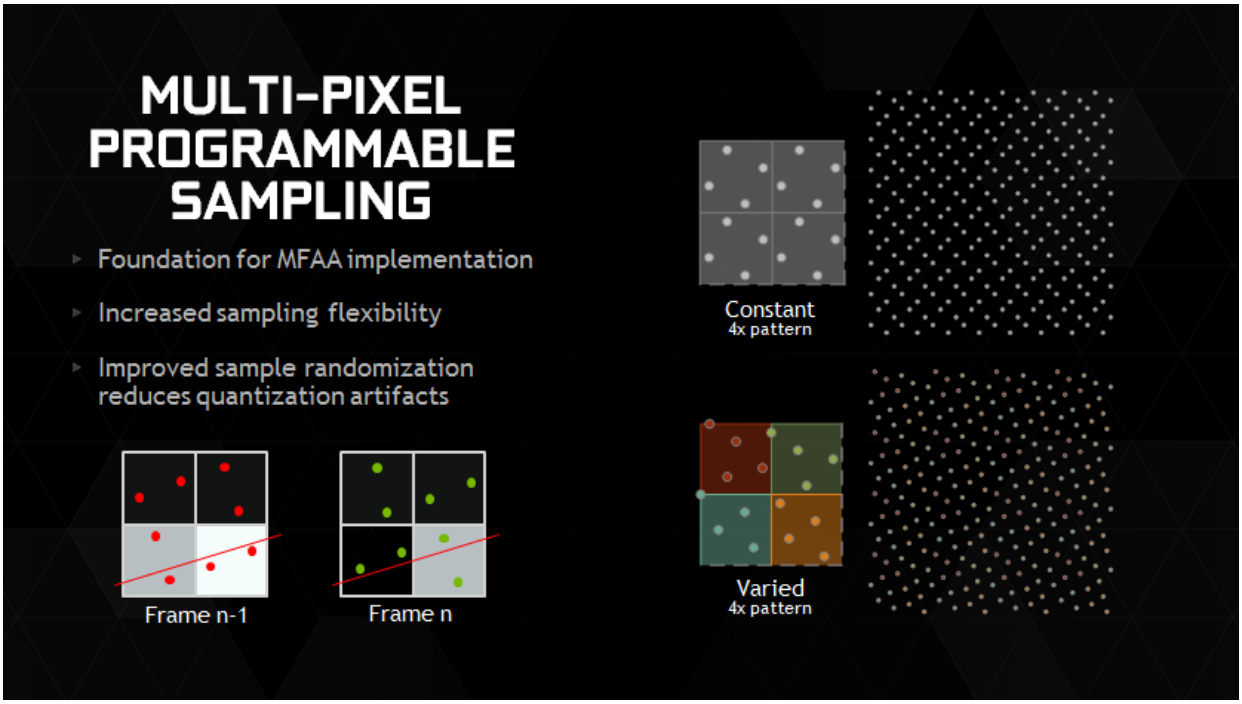
Spanning devices ranging from PCs to consoles, tablets, and smartphones, Microsoft's upcoming DirectX 12 API has been designed to have CPU efficiency significantly greater than earlier DirectX versions. One of the keys to accomplishing this is providing more explicit control over hardware—giving game developers more control of GPU and CPU functions. While the NVIDIA driver very efficiently manages resource allocation and synchronization under DX11, under DX12 it is the game developer's responsibility to manage the GPU and GPU. Because the developer has an intimate understanding of their application's behavior and needs, DX12 has the potential to be much more efficient than DX11 at the cost of some effort on the part of the developer. DX12 contains a number of improvements that can be used to improve the API's CPU efficiency; we've announced that all Fermi, Kepler, and Maxwell GPUs will fully support the DX12 API.

In addition, the DX12 release of DirectX will introduce a number of new features for graphics rendering. Microsoft has disclosed some of these features, at GDC and during NVIDIA's Editor's conference. Conservative Raster, discussed earlier in the GI section of this paper, is one such DX graphics feature. Another is Raster Ordered Views (ROVs,) which gives developers control over the ordering pixel shader operations. GM2xx supports both Conservative Raster and ROVs. The new graphics features included in DX12 will be accessible from either DX11 or DX12 so developers will be free to use these new features with either the DX11 or DX12 APIs on GPUs that implement the features in hardware.

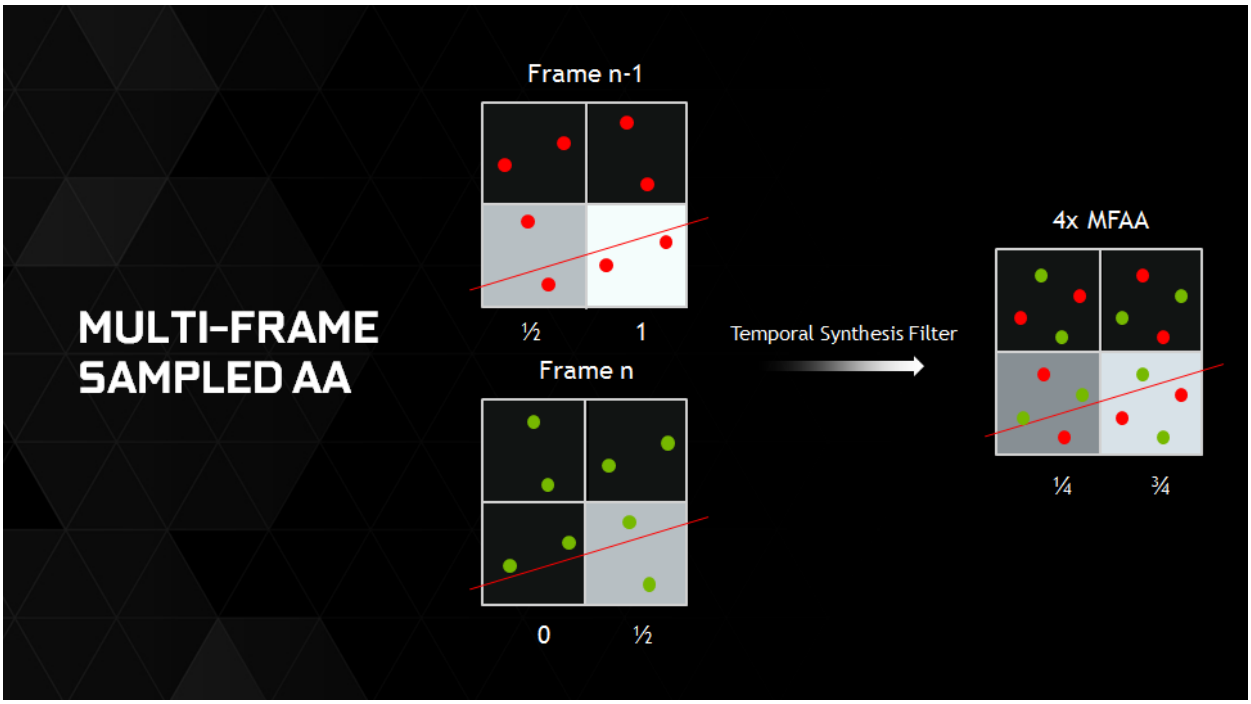
Advancing the State-Of-The-Art in Image Quality

Game developers and GPU vendors are increasingly implementing more advanced forms of anti-aliasing (AA) to enhance image quality. GM2xx GPUs have a number of new features for much more flexible sampling, enabling further advancements in AA quality and efficiency.

Today's GPUs ship with fixed sample patterns for AA that are stored in ROM. When gamers select 2x or 4xMSAA for example, the pre-stored sample patterns are used. While many current games implement deferred, post-processed AA techniques such as FXAA or SMAA, there are still many others that continue to use traditional hardware-based multi-sample AA (MSAA). GM2xx GPUs support **multi-pixel programmable sampling** for rasterization, providing opportunities for more flexible and novel AA techniques to be implemented in the context of both deferred and conventional forward rendering.



With programmable sample positions, the ROMs that were used to store the standard sample positions are replaced with RAMs. The RAMs may be programmed with the standard patterns, but the driver or application may also load the RAMs with custom positions which are free to vary from frame to frame or even within a frame. In a 16x16 grid per pixel, we have 256 different locations to choose from for each sample. We've also extended this programmable sample location support to span multiple pixels, so for example in 4x MSAA rendering, all 16 samples within a 2x2 pixel footprint can be located arbitrarily. This sample randomization can greatly reduce the quantization artifacts (like stair-stepping) that occur with more traditional forms of AA. These freely specified sampling positions may be used in the development of effective new algorithms.



NVIDIA engineers have recently developed new AA algorithms that vary, in interleaved fashion, the sample patterns used per pixel either spatially in a single frame (where, for example, each successive pixel uses one of four different 4xAA sample patterns) or interleaved across multiple frames in time. **Multi-Frame Sampled AA (MFAA)** is a new AA technique that alternates AA sample patterns both temporally and spatially to produce the best image quality while still offering a performance advantage compared to traditional MSAA. The final result can deliver image quality approaching that of 8xAA at roughly the cost of 4xAA, or 4xAA quality at roughly the cost of 2xAA.

MFAA is still under development. We'll provide more details on the technology once we get closer to release.

Dynamic Super Resolution

Thanks to rapidly falling LCD prices, the popularity of 4K displays has surged this year. But not all gamers want to spend the money on a new monitor. For the eye candy purists who don't want to splurge but want to approximate the crisper visuals of a 4K panel, many have turned to the process of "downsampling." This is where the GPU renders the game at a resolution higher than the screen can display, and then scales the image down to the native resolution on output to the user's display.

Downsampling require users to set up custom displays with the graphics driver control panel, and adjust various low-level display parameters to appear properly. So it's not necessarily the friendliest way to improve image quality. Also, while downsampling can provide a significant improvement in image quality, artifacts are sometimes observed on textures and when certain post-processing effects are applied.

To address the usability and quality issues, NVIDIA has developed a method called **Dynamic Super Resolution**. In principal, Dynamic Super Resolution works like traditional downsampling, but it has a simple on/off user control, and it uses a 13-tap Gaussian filter during the conversion to display resolution. The high-quality filter reduces or eliminates the aliasing artifacts experienced with the simple downsampling, which relies on a simpler box filter.

Note that people often confuse downsampling (and now Dynamic Super Resolution) with the traditional Supersampling method of anti-aliasing. All three techniques render at higher resolutions internally, and then filter down to lower resolution for output. The difference is that downsampling and Dynamic Super Resolution actually have the game render at the higher resolution, so the game believes it's running on a higher resolution display, and the GPU then filters and samples down. The process should work with

most games well, aside from some issues with visibility of onscreen game controls being displayed on lower resolution monitors. With supersampling, the game still renders at a particular resolution—say 1920x1080—and the GPU upsamples that resolution without the game's knowledge, and then filters back down. This can cause issues with newer games that use post-processing effects, or are expecting the full rendering process to be at a given resolution set by the game itself.

Figure 15: *A screenshot from Dark Souls 2. Standard 1080p on the left, DSR on the right*

Dynamic Super Resolution can be found in the control panel of our Release 343 driver, as well as GeForce Experience, where we provide Optimal Playable Settings (OPS) for Dynamic Super Resolution for today's hottest games. While it's compatible with all GeForce GPUs, the best performance can be seen when using a GeForce GTX 980. Going forward we could potentially use Maxwell's more advanced sampling control features, like programmable sample positions and interleaved sampling, to further improve Dynamic Super Resolution for owners of GM2xx GPUs.

Conclusion

GeForce GTX 980 was designed to deliver extreme gaming performance with unprecedented levels of power efficiency. It's the fastest GPU we've ever built, yet its TDP is just 165 watts. The Maxwell GPU inside the GeForce GTX 980 is built on the same 28-nm manufacturing process used for previous GeForce GPUs—this means that the GPU's remarkable performance and efficiency doesn't come from the smaller transistors you get with a die shrink, rather these benefits come from the new GPU architecture.

Maxwell GPUs feature a completely revamped chip topology that's been optimized to provide record-breaking performance per watt. The Maxwell SM is grouped into four separate CUDA core processing blocks, each with its own dedicated resources for scheduling and instruction dispatch. With double the PolyMorph Engines, geometry and tessellation performance is significantly improved. Finally, with double the ROPs, and enhanced compression algorithms, the GeForce GTX 980 delivers exceptional performance with MSAA.

When it comes to AA, the GeForce GTX 980 has been tailored to offer the crispest visuals available on the PC. The GeForce GTX 980 supports new features for sampling control that will enable new AA techniques like MFAA, allowing lower level AA sample patterns to be perceived as higher quality AA, but with the faster performance of lower AA levels. And the GeForce GTX 980 supports Dynamic Super Resolution technology, an NVIDIA-developed version of downsampling that brings 4K visuals to existing 1080p displays without the configuration hassles or IQ bugs that currently exist with downsampling today.

Real-time global illumination as seen in Epic's UE4 'Elemental' tech demo has been on the wish list of game developers for years, but has remained impractical—Epic chose to remove their Sparse Voxel Octree Global Illumination (SVOGI) implementation to produce the 'Infiltrator' tech demo that was released the next year. With VXGI, the GeForce GTX 980 can render global illumination radically faster than prior GPUs, making the lighting seen in the Elemental tech demo more feasible for the next generation of PC games.

Offering built-in support for HDMI 2.0, the GeForce GTX 980 is ready for gaming on the latest big screens.

Thanks to its revolutionary Maxwell architecture, the GeForce GTX 980 is the perfect upgrade for gamers who want to enjoy the hottest new games shipping at the end of 2014 and well into the future. It's the most advanced GPU ever made.

LEGAL NOTICE