Intel® Processor Graphics Gen11 API Developer and Optimization Guide

Version 1.0

1. Introduction

This document presents developer guidance and optimization methods for the graphics hardware architecture of Intel® Processor Graphics Gen11. It provides developers best practices to most effectively harness the architecture's capabilities and peak performance. The document also provides specific API guidance for using the latest graphics APIs on Intel Processor Graphics Gen11.

The intended audience of this guide is developers who seek to optimize their interactive 3D rendering applications for Gen11. It is assumed that the developer has a fundamental understanding of the graphics API pipelines for Microsoft DirectX* 12, Vulkan*, and/or Metal 2. Gen11 also supports the DirectX 11 and OpenGL* graphics APIs; however, there are performance benefits and lower CPU overhead for applications that use the newer and lower level APIs such as DirectX12, Vulkan, and Metal 2, and also new graphics architecture features that are only available in these APIs.

2. Intel® Gen11 Architecture Highlights

Gen11 offers improved performance and efficiency over Gen9, and new features such as coarse pixel shading, tile-based rendering, and new display controller features. In addition, Gen11 offers the following improvements over previous generations:

- Compute capabilities that deliver up to a teraflop of performance
- Significantly lower shared local memory (SLM) latency
- Larger L3 cache size
- Increased memory bandwidth
- Improved multisample anti-aliasing (MSAA) performance

For a more in-depth overview of the Gen11 architecture and new features, see the *Intel® Processor Graphics Gen11 Architecture* guide.

2.1 Tile-Based Rendering

Gen11 implements a tile-based rendering solution known as position only shading tile-based rendering (PTBR). The motivation of tile-based rendering is to reduce memory bandwidth by efficiently managing multiple render passes to data per tile. In order to support tile-based rendering, Gen11 adds a parallel geometry pipeline that acts as a tile binning engine. It is used ahead of the render pipeline for visibility binning pre-pass per tile. It loops over geometry per tile and consumes visibility stream for that tile. PTBR uses the L3 cache to keep per tile data on die, reducing external memory bandwidth. For more information, refer to the architecture guide or talk with an Intel application engineer to see if your workload will benefit.