The Trend of Online Gaming Platform and Developing Considerations

Senior Application Engineer, Intel Corp. | Mark Kim (mark.kim@intel.com)

1. Introduction

Gamers are the most early-adaptor on newest/enthusiastic PC platform. There are several trends on gaming platform for online game play - Multi-core platform became a trend of PC and notebook platform for gamers, and notebook computer began to be considered as suitable platform for gaming according to the performance enhancement, and high-end graphic card. Gamers who bought a new platform want to experience more benefit on their own platform? better frame rate for flawless rendering, better effect with highest option available in the gaming application.

To satisfy those gamers, gaming developer should consider multi-threading to get a benefit of multi-core, and more feature for high-end platform, and optimization work to get a benefit on the newest platform. But still multi-threading and optimization work are not familiar to many of gaming client developers? most of gaming client managing 1 main thread, and no serious considering for the optimization work due to the tight development schedule.

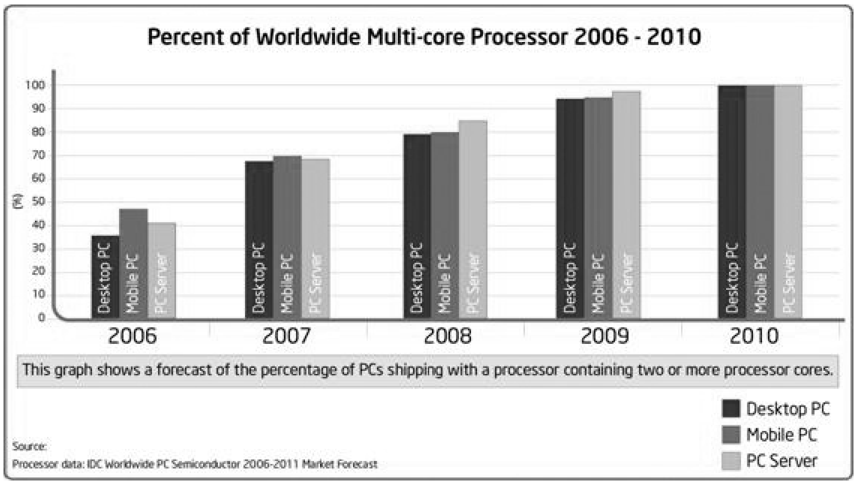
In this article, I' ll deal with the issues on managing multi-thread on gaming client application and best practices to give hints to the client developers. Also I' ll introduce the tools and methodology to optimize gaming client. The reason why integrated graphic should be considered as one of main graphic card for gamer will be illustrated and I' ll introduce the way of optimization way for the integrated graphics.

2. Trend of Gaming Platform

2.1. Multi-core

Multi-core processors are becoming the computing standard ? servers, PC' s and consoles. By 2009, almost all servers, laptops and desktop PC' s will have multi-core processors.

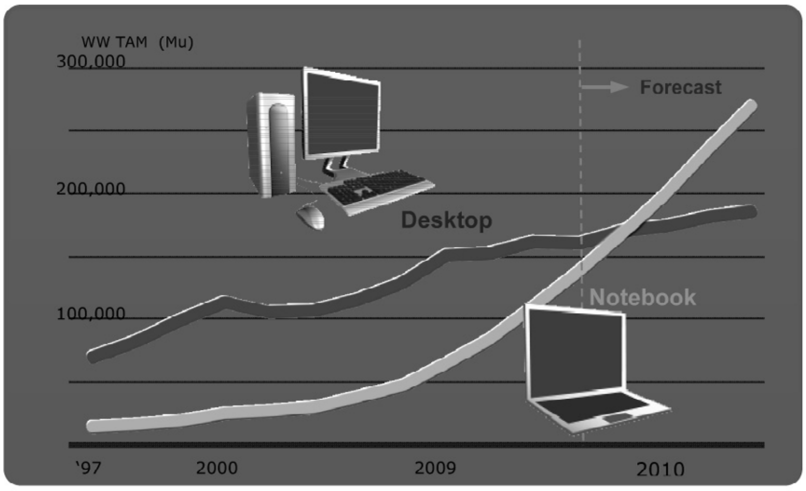
Figure 1 Percent of Worldwide Multi-core Processor 2006-2010



2.2 . Notebook

Notebook adoption is set to surpass Desktops by 2009 (earlier in some geographies). According to the fast performance enhancement with the enhanced performance of integrated graphics, notebook will be another main gaming platform.

Figure 2 Worldwide Notebook and Desktop TAM 1997-2010

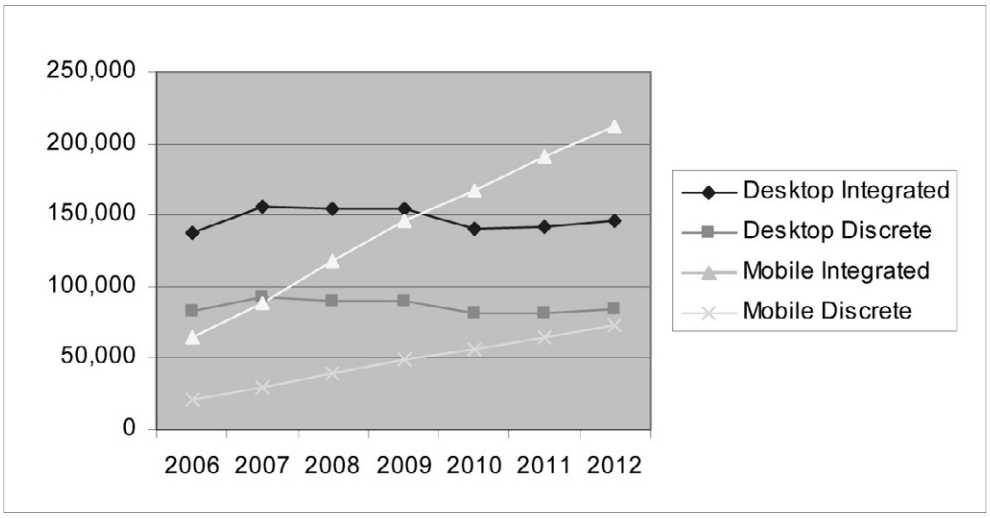


Source : Gartner December 2007

2.3 Graphics

There are two lines of graphic cards currently ? external graphics and integrated graphics. Most of gamers adapted external graphic card because of the performance requirements of most of 3D games. But many of casual games and 2D games runs well on integrated graphics, and moreover, Intel plans to announce new integrated graphics with 'playable' performance for the many of 3D MMORPG games, and supports DirectX10. The two lines of graphic card will be go on, but integrated graphics should be considered as important as an external graphics because the market share of internal graphics is expected to grow faster than external graphics.

Figure 3 Graphic Card Market Share 2006-2012



Source : Mercury Research (Q4'07)

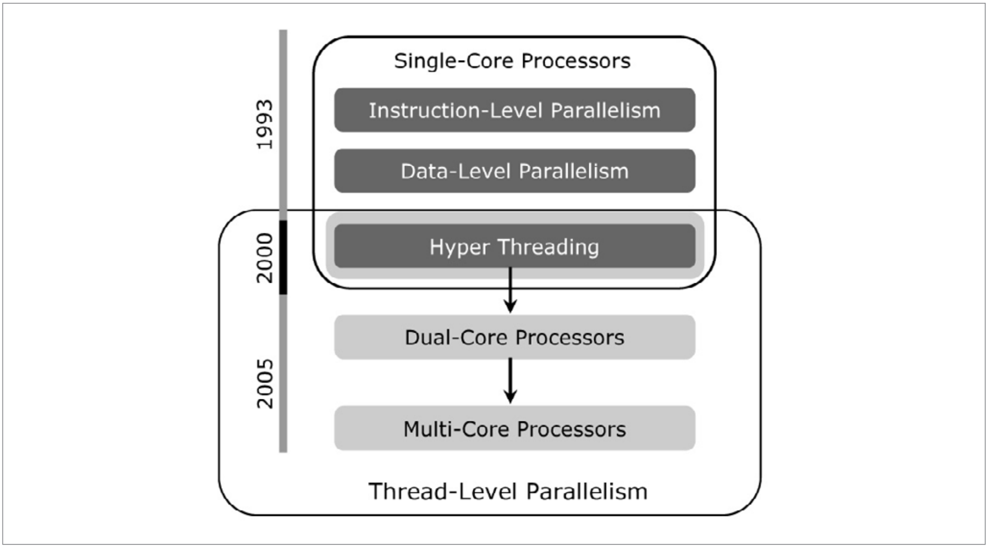
3. Developing Considerations

3.1. Multi-core

3.1.1. History of parallel architecture

In the early days, parallelism was enabled by splitting instructions into multiple streams that multiple processors executed. In 2002, Intel introduced Hyper-Threading Technology (HT Technology), which provides thread-level parallelism on single-core processors. HT Technology enables a single processor to simultaneously run two programs, as long as the next instruction from each program did not need to use the same functional units. HT Technology paved the way for the next major innovation, dual-core platforms. In HT Technology, two threads share the same processor. If the threads require execution units simultaneously, each thread' s flow of instructions would affect the other. Therefore, it may be a better idea to run threads on separate cores. In 2005, Intel introduced dual-core processor?based platforms, which contain two fully processing cores on a single chip, with a dedicated L1 cache and a shared L2 cache.

Figure 4 Evolution of Multi-Core Processors



Source : Mercury Research (Q4' 07)

The next logical step was quad-core processors. The first version of the quad-core processors combined two dual-core processors. The generic name of processors containing more than one core is multi-core processors. At present, multi-core processor-based platforms contain Intel processors with four cores and may soon evolve to include more cores.

3.1.2. Thread programming

Multithreaded programming uses threads to concurrently execute multiple operations. It focuses on the design, development, and deployment of threads within an application and the coordination maintained between the threads and their respective operations. It is important to determine the threading strategy in the design phase itself, while developing an application. In this phase, you would consider all data and code restructuring related to threading; this will help minimize the development effort by minimizing the possibility of redesign. You may thread your application for various reasons: reducing the execution time of your application, increasing the throughput, or offsetting the time taken by slower operations in the application. Depending on your requirement, you can decide which threading strategy you would employ for your application. There are two types of method to decompose and allocate resources to each thread.

Task or functional decomposition refers to decomposing a program based on the different computations it performs.

Domain decomposition refers to dividing large data sets, whose elements can be computed independently. Along with each chunk of data assigned to a thread, the associated computation on that chunk is also assigned to the thread. Typically, the same independent operation is applied repeatedly to each of the different data elements.

3.1.3. Programming trend - Multi-threaded gaming client

Traditionally, most of online gaming client managed only 1 main thread (even though there are many of threads which are managed by DirectX or other libraries. Most of client developers were not familiar with multi-thread programming environment - comparing with server developer. From 2005, Intel began to work with Korea gaming client engineers to thread their clients.

There are 2 way of taking a benefit from multi-core by adapting multi-threaded architecture. The one is enhancing performance and the other is adding new feature with remaining resources. I' ll introduce several multi-threading works adapted to real applications.

a. Data decomposition

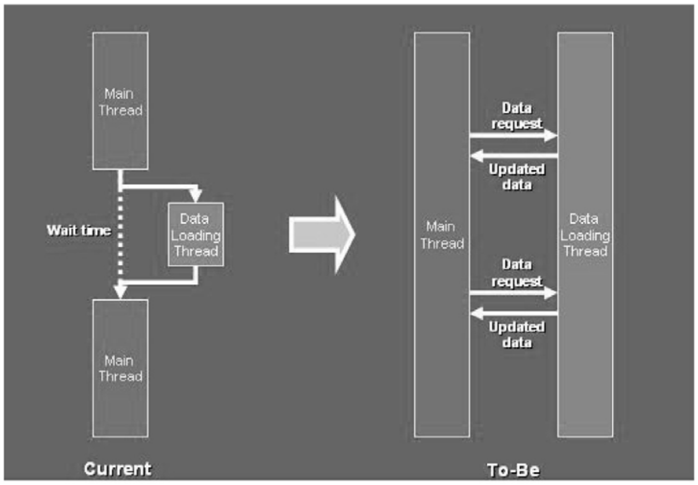
Legacy & published games are not easy to change their program architecture to adapt multi-thread on their application. In that case, developer can consider adapting 'Domain decomposition' to the hotspots if the module deal with enough size of data and the data can be calculated concurrently. OpenMP can be a best solution in this case.

b. Separating data loading thread

If a MMORPG game using seamless world, it can shows data loading lag when character moves to another position. (New data should be loaded for many case of character movement)

That is held on single-core, and dual-core machine, because the main thread waits until data loading thread finishes data loading, and it is a kind of bad experience for gaming users.

Figure 5 Separating data loading thread from main thread

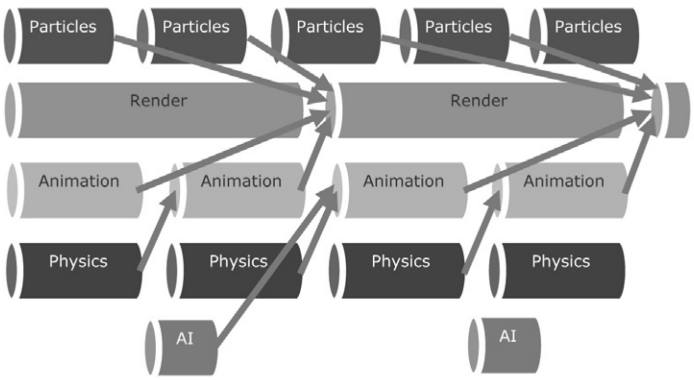


It is caused by managing 1-thread at a time, and the other thread should wait until other thread is ended. They uses thread program, but not a real multi-threading (that meant only 1 thread runs at a time). We' ll solve this problem by managing 2 main-threads (for main and for data-loading) at a time. When there is no updated data or requested data is not ready, and then main thread will skip environment update (or automatically update environments).

c. Adding physics & effect

Physics and effects consumes lots of CPU resource and so it can be targets of multi-threading. Still online game is lack of physics and effects comparing with PC games.

Figure 6 Maintaining physics and effect threads



d. Using multi-threaded gaming engine

If a gaming client adapted a 'multi-threaded' gaming engine such as 'Unreal3' or 'Crysis', it can get a benefit on multi-core platform automatically. Several games in Korea began to adapt those engines from 2007. If a gaming client separate more threads adding to those threads maintained by the gaming engine, then it can get a benefit more than 2 cores.

3.1.4. Technical issues

There are 2 issues regarding to managing threads - Correctness and Performance.
Correctness: Creating and managing threads explicitly increases the complexity of code. Managing simultaneous activities and their interactions can result in problems related to synchronization and communication. Therefore, it is important to check threaded applications for issues of correctness such as storage conflicts, deadlocks, and execution order. The issues related with thread correctness can be detected by tools such as Intel® Thread Checker™

Performance: In addition to considering design and correctness concepts while threading applications, you can measure the performance improvement of multithreaded execution over serial execution. The performance concepts to consider while threading applications are speedup, efficiency, granularity, and load balance. Performance issues on threaded application can be analyzed with help of tools such as Intel® Thread Profiler™

3.2. Vectorization

3.2.1. Meaning Vectorization

Vectorization means 'separate data to SIMD instructions deal with the data'. Vectorization can be one of the best ways of optimization especially for the application which calculates tons of independent data.

3.2.2. History

SIMD computations were introduced to the architecture with MMX technology. MMX technology allows SIMD computations to be performed on packed byte, word, and doubleword integers. The integers are contained in a set of eight 64-bit registers called MMX registers (see Fig)

The Pentium III Processor extended the SIMD computation model with the introduction of the Streaming SIMD

Extensions (SSE). SSE allows SIMD computations to be performed on operands that contain four packed single-precision floating-point data elements. The operands can be in memory or in a set of eight 128-bit XMM registers (see Fig). SSE also extended SIMD computational capability by adding additional 64-bit MMX instructions.

Below is the brief history of Vectorization technology supported by each generation of Intel CPUs.

MMX™ technology - Intel® Pentium® with MMX™ and Pentium® II processors

- Introduced 64-bit MMX registers for SIMD integer operations
- Supports SIMD operations on packed byte, word, and double-word integers
- Useful for multimedia and communications software

SSE - Intel® Pentium® III processor

- Introduced 128-bit extended memory manager (XMM) registers for SIMD integers and FP-SP operands
- Executes FP and SIMD simultaneously
- Introduced data prefetch instructions
- Useful for 3D geometry, 3D rendering, and video encoding/decoding

SSE2 - Intel® Pentium® 4 and Intel® Xeon™ processors

- Added extra 64-bit SIMD integer support
- Has same XMM registers for SIMD integer and floating point double precision (FP-DP)
- Has 144 new instructions for data support (no new registers)
- Adds support for cacheability and memory ordering operations
- Useful for 3D graphics, video encoding/decoding and encryption

SSE3 - Intel®Pentium® 4 Processor

- Accelerates performance of Streaming SIMD Extensions technology, Streaming SIMD Extensions 2 technology, and X87-FP math capabilities.
- Useful in some 3D operations (Quaternions), complex arithmetic and video codec algorithms

SSSE3 - Intel® Core™ 2 Processor

- Introduces 32 new instructions to accelerate eight types of computations on packed integers.

SSE4.1 - Intel®Core™ 2 Processors (Penryn based)

- Introduces 47 new instructions to accelerate video, imaging and 3D applications. SSE4.1 also improves compiler vectorization and significantly increase support for packed dword computation.

3.2.3. Adaptation to the real application

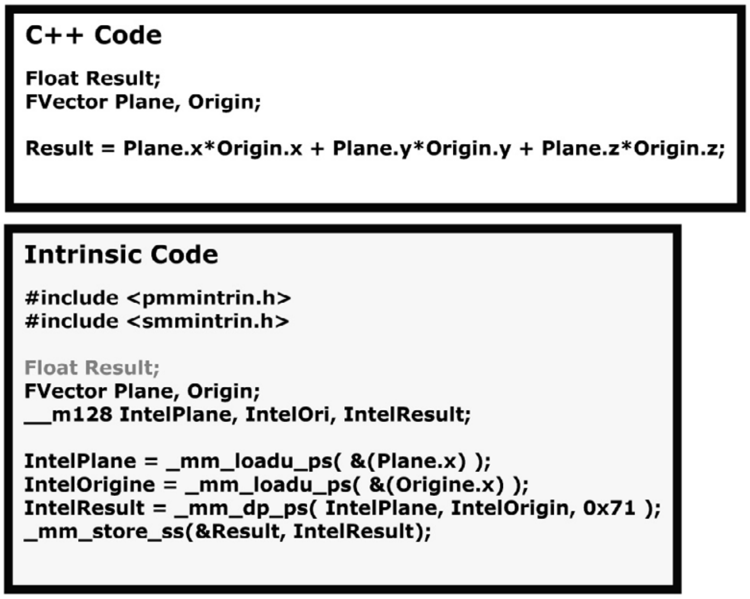
- Vectorization by Intel Compiler
  Intel compiler supports newest SIMD instruction sets for the newest CPUs. By using suitable switch, developer don' t need to add or modify their source code to support SIMD, instead of that, developer can add proper compiler switch which enables auto-vectorization suitable for the CPUs regarding to the compiler switches. (Compiler switch and description about compiler based vectorization)

Usually, such a big application as most of popular online game is not easy to adapt Intel compiler instead of other compiler. It is related with compatibility and standardization. (Intel supports standard, but some of other compiler use their own library and keywords.

To make the problem simple, developers can adapt Intel compiler for the specific projects or files.

- Intrinsic code
  Even though vectorization by compiler can be easiest way, most of application can' t be vectorized because of several reasons including 'data dependency' . In that case, developer should vectorize the data for most suitable SIMD instruction. Intrinsic code is a middle layer language matched with each SIMD instructions which is easier to be understood than assembly language.

Figure 7 C++ code and corresponded intrinsic code



C++ Code

```
Float Result;
FVector Plane, Origin;

Result = Plane.x*Origin.x + Plane.y*Origin.y + Plane.z*Origin.z;
```

Intrinsic Code

```
#include <pmmintrin.h>
#include <smmintrin.h>

Float Result;
FVector Plane, Origin;
__m128 IntelPlane, IntelOri, IntelResult;

IntelPlane = _mm_loadu_ps( &(Plane.x) );
IntelOrigine = _mm_loadu_ps( &(Origine.x) );
IntelResult = _mm_dp_ps( IntelPlane, IntelOrigin, 0x71 );
_mm_store_ss(&Result, IntelResult);
```

3.3. Optimization on Integrated graphics

3.3.1. General graphic optimization

Optimizing for graphics performance in general will make IIG performance better. Generalizations that you hear for discrete graphics cards apply just as well to integrate. There are several possibilities for optimization works.

- Batch Primitives
- Reduce state changes
- Use smaller textures when possible
- User PURE DX devices
- Smaller shaders perform better
- Prefer static over dynamic geometry
- Never query the GPU if you can help it

4. Conclusion

Gaming platform for playing online games evolved to multi-core and high-end graphics and notebook computer. Gamers who bought new platform wants to experience better experience on their platform. Multi-threading is not an optional to client developer but a mandatory to give best experience to gamers. Game developer can give more benefit to the multi-core users with enhanced frame rate or added features. Verification and performance optimization on threaded application can be easier with help of analyzing tools such as Intel® Thread Checker™ and Intel® Thread Profiler™ . Vectorization can be one of the most effective ways of performance optimization and it can be easily done by compiler

김 준 호 Mark Kim (mark.kim@intel.com)

- Senior application engineer, Software Solution Group, Intel Corp.
- MS. Computer Science, KAIST
- BS. Computer Sceince, Yonsei Univ.