

SESSION: 2018/19 DIET 1

Module Title: Games Programming 3

Module Code: MHI622946

Level: 4

Module Leader: Dr. Richard Holden

Individual Marks Summary:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Component | A: Environment set-up | B: Game Coding | C: Persistence | D: Report |
| Component % | 5 | 40 | 15 | 40 |

|  |  |  |
| --- | --- | --- |
| Student Name | Student ID | Student E-mail |
| Gary Mulhall | S1626456 | Gmulha200@caledonian.ac.uk |

Student Declaration

This piece of work is not plagiarized. It is my own original work and has not been submitted elsewhere in fulfilment of the requirements of this or any other award.

Signature: Gary Mulhall Date:13/12/2018

Table of Contents

[What is Moore’s Law? 3](#_Toc532556478)

[The End of Moore’s Law 3](#_Toc532556479)

[Multi-Core Technology 3](#_Toc532556480)

[Parallelism in Multi-Core Technology 3](#_Toc532556481)

[Differences in Parallelism 4](#_Toc532556482)

[Multi-Threaded Technology 4](#_Toc532556483)

[Relevance to PopStarWest 4](#_Toc532556484)

[In Conclusion 5](#_Toc532556485)

[References 6](#_Toc532556486)

# What is Moore’s Law?

Moore’s Law is an observation made over 50 years ago that the performance of integrated circuits will double every year due to transistors becoming smaller (Mack, 2011), meaning that it would be possible to fit chips with twice as many transistors per year. This observation has been strikingly accurate and has been used to calculate future trends (Schaller, R.R., 1997.) for half a century.

# The End of Moore’s Law

Moore’s Law is now reaching a point where it can no longer be used to calculate future trends. Technologically, we are reaching a point where transistors have been decreased to such small sizes that it is now almost impossible to make them any smaller, thus meaning it is impossible to double the amount per chip. A recent article by (Waldrop, 2016) touches on the fact that adding more and more silicon to these very small chips every year leads to a drastic increase in heat, which in turn, leads to degradation of performance. Waldrop also mentions in the same article that Moore’s Law was not an inevitability and it seemed more likely that hardware manufacturers were intentionally sticking to the observation. In recent years, large hardware manufacturers like Intel have been adding multiple cores to each of there chipsets in an attempt to maintain the demands of modern software.

# Multi-Core Technology

Multiple cores have been used in central processing units in recent years. This means that the processors contain multiple execution cores, which in turn means that groups of transistors can be split across the cores and be used to complete separate task in parallel with each other. This increases efficiency immensely as it reduces stress on the chipset by spreading the load, power consumption is also reduced greatly. Multi-core chips do not always exceed single cores in processing speed (Geer, 2005), but they do increase the overall efficiency and are seen as highly beneficial by many due to this fact.

# Parallelism in Multi-Core Technology

Single core technology will typically execute tasks in a sequential order, meaning tasks are completed one-by-one in a linear fashion until there are no more tasks to execute. Multi-core programming allows this to be changed up and introduces the possibility of parallelism. Parallelism is used to execute multiple tasks all at once (Gepner and Kowalik, 2006) by spreading the task load across the multiple available cores. Many developers believe that parallel programming for multiple cores must become more common practice to ensure multi-core processors are being used to their fullest potential, and to meet the ever-increasing demands of modern software.

# Differences in Parallelism

The two main types of parallelism used in multi-core developments are task parallelism and data parallelism. Task parallelism is used to fully take advantage of the multiple cores available on the processing chip by executing multiple separate tasks across these cores in parallel. Synchronized and asynchronous task parallelism are different execution methods and are down to development choices or needs for the program. Synchronized task parallelism is used to ensure that all tasks are completed and finalized within a single clock cycle, whereas asynchronous parallelism will allow tasks to execute and run for as long as is required without forcing them to end.

Data parallelism, however, is used to perform a single task using all multiple cores all the while pulling from a shared source to ensure the task is being completed efficiently. Data parallelism does drastically increase efficiency as tasks can be completed much quicker but does limit the number of tasks that may be completed at once.

Each method has its pros and its cons and should be used depending on the developer and user requirements.

# Multi-Threaded Technology

Multi-threading can be used in tandem with multi-core programming and allows for the multiple threads in a processing unit to break a task down into smaller chunks which can be executed by each. This will decrease computation times and will help efficiency overall, however, must be approached with caution as it can cause issues when threads are altered multiple times by accident etc.

This is where the real current problems with multi-core and multi-thread development lie, developers do not have the understanding of how to develop effectively and efficiently in this way and some appear to have a reluctance to try.

# Relevance to PopStarWest

The contract with PopStarWest has requested the development of a game with many features, some of which are very similar and require creation of many of the same asset. With this in mind, this section will aim to explain the advantages of multi-core programming in game development and how it would be beneficial to PopStarWest.

As previously discussed, programming for the exploitation of multiple cores is guaranteed to bring overall efficiency increases for developers and users when done correctly. Methods like scalable parallelism, in which large tasks and broken down into smaller tasks and spread across cores, can be game changing. This is very true when it comes to video game development as it also means that a game can have much more content without lagging or struggling to complete computations. Take for example the mushrooms in the game PopStarWest have requested. These mushrooms are all the same with slightly randomized features (rotation speed etc.) meaning that the creation of the mushrooms could potentially be spread across multiple cores using scalable parallelism, which also means that more mushrooms could be spawned without a degradation in quality. This would increase the gameplay experience vastly by changing how tasks are handled.

Handling things like the physics and collisions of said mushrooms (or any physics-based object) using multi-threading on multiple cores will increase efficiency even more and will reduce any stutters or loss in framerate when physics-based collisions are made. Also, things like particle effects, which can be extremely expensive computationally when happening multiple times in a scene, can be handled using scalable parallelism and spreading the effect across multiple cores. A small change for a huge addition.

El Rhalibi, Merabti, and Shen (2006) believed that with the leaps and bounds that video games were making at the time, developers must discard the single loop programs they had been using almost exclusively up to that point. This belief was based on the fact that games were becoming increasingly large in scope and a new approach must become the norm to continue that growth.

# In Conclusion

When thinking about current and future game development in particular, the single execution loop will simply not cut it if innovation is to continue. Programming specifically for the purpose of exploiting multi-core processors and threads must become the norm to ensure video games continue their growth. Learning how to make the most of the different parallelism methods would be a great first step in increasing optimization in video games. Users have become used to seeing video games become more impressive, year on year, with more features and better graphics, higher populations in game and more abilities and effects. This has been possible to a point thanks to Moore’s Law being a good basis for estimations in how far technology could be pushed. However, as we move away from Moore’s Law and accept the end of the law, the investment must be made by companies and developers to ensure programmers have the necessary skills and understanding to develop using multiple cores and threads.

This will bring with it many issues at first, as it is not as simple as learning to use a new piece of software may be but changing the way that development has been approached for so many years. It will however ensure video games grow and continue to be an important part of many peoples lives and a highly profitable part of the entertainment industry.

# References

El Rhalibi, A., Merabti, M. and Shen, Y., 2006, April. Improving game processing in multithreading and multiprocessor architecture. In *International Conference on Technologies for E-Learning and Digital Entertainment* (pp. 669-679). Springer, Berlin, Heidelberg.

Geer, D., 2005. Chip makers turn to multicore processors. *Computer*, *38*(5), pp.11-13.

Gepner, P. and Kowalik, M.F., 2006, September. Multi-core processors: New way to achieve high system performance. In *Parallel Computing in Electrical Engineering, 2006. PAR ELEC 2006. International Symposium on* (pp. 9-13). IEEE.

Mack, C.A., 2011. Fifty years of Moore's law. *IEEE Transactions on semiconductor manufacturing*, *24*(2), pp.202-207.

Schaller, R.R., 1997. Moore's law: past, present and future. *IEEE spectrum*, *34*(6), pp.52-59.

Waldrop, M.M., 2016. The chips are down for Moore’s law. *Nature News*, *530*(7589), p.144.