# An Effective way to optimize performance and battery life of smart devices using multi-kernel approach

Sadid Rafsun Tulon
School of Computer Science
Carleton University
Ottawa, Canada K1S 5B6
*stulo080@uottawa.ca*

November 30, 2021

**Abstract**

A very concise summary of the problem addressed and solution presented in this paper.

## 1 Introduction

Smart devices have become an integral part of our life. Many of the smart devices are battery-powered. The current battery technology is the biggest handicap of portable smart devices right now. As CPUs and system on a chip(SoC)'s are improving their efficiency, things are getting better. However, modern software also demands more processing power which drains the battery faster. The lightweight kernels can extend battery life much further, but their software compatibility is far from ideal. Now that battery technology reached a plateau and transistors in the semiconductors are also close to reaching their size limit, we are in need of an alternate way of achieving better battery life without compromising performance and compatibility. A new approach using multiple operating system (OS) kernels can help to mitigate this problem. Almost all modern chips are multi-core. We can use these multi-core chips to run multiple OS kernels at the same time. Ideally one of the kernels is a lightweight kernel for better performance and battery life, and a monolithic kernel for better compatibility. The system can switch between the kernels, run them simultaneously, or stop them from running depending on the need. This way we will be able to get better battery life while still retaining the software compatibility. This research is an effort to optimize performance, compatibility, and battery life by utilizing a multi-kernel approach while using existing battery technology.

In this project, the idea is to research if we fit two operating systems for both cores of multi-core processors individually and how they perform.

## 2 Literature Review

The current hardware technology is changing rapidly. The software is struggling to keep up with this rapid change . Thus it has become very challenging to optimize software for a specific hardware[1] . So we need a new OS structure which can be as scalabe as needed. Most modern system use multicore processor. We can leverage this to our advantage.

Light Weight Kernels (LWKs) are not new. They are here for at least 30 years [6]. But they are only gaining popularity in these past couple of years because of High-Performance Computing (HPC). HPC requires extreme scalability which the complex Full-Weight Kernel (FWKs) are very slow in adapting. LWKs work extremely well with highly scalable systems as they have a simple codebase. Moreover, they have better performance and efficiency than FWKs. However, LWKs only devices would have extremely limited functionality as they have limited compatibility with the standard APIs [4], [3] . Another major issue of LWKs are driver support. As LWKs have simple codebase mainly focusing on perfromance, they are missing many advance features that FWKs have. [4] In order to achive more performance and efficiency out of the system without sacrificing software and hardware compatibility, multikernel operating systems have been proposed. Multikernel OS runs both FWK and LWK side by side divides their tasks.[performance and scalabily] The LWK usually responsible for the high performance tasks and FWK is there for ensuring compatibility, driver support and also as a backup system.[4]. This system works fairly well in high performance super computers.

Figure 2 shows the system architecture of a multikernel system.

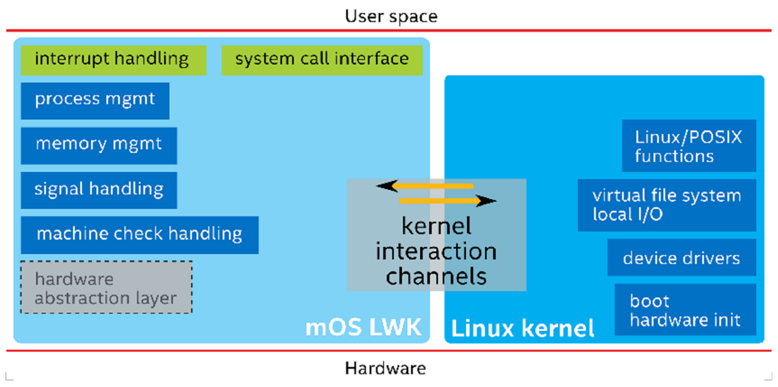

Figure 1: System Architectural Overview [6][2][5]

In this paper we will explore if we can use this multikernel method on the battery powered smart devices. We will compare performance and power efficiency between single linux kernel system and multikernel system. As batteries have very small capacity, ensuring longer battery life is the most import part of this project. Thus power efficiency will be one of the main focus of this research project.

## 3 Problem Statement

1. A concise statement of the question that your paper tackles.
2. Justification, by direct reference to your Literature Review, that your question is previously unanswered.
3. Discussion of why it is worthwhile to answer this question.

## 4 Proposed Solution: ...

This part of the paper is much more free-form. It may have one or several sections and subsections. But it all has only one purpose: to convince the reader that you answered the question or solved the problem that you set for yourself. In this section you will for example present new algorithms you developed and your implementation of these new algorithms.

Figure 2 is an example of a drawing created with *mdraw* or *epsfig*.

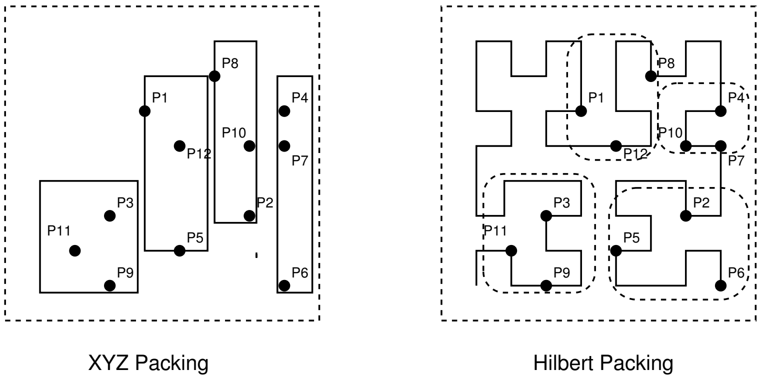

Figure 2: XYZ and Hilbert Packings

## 5 Experimental Evaluation

This section is not mandatory for all papers (for example theory papers) but typically required for papers in the field of parallel computing. After all, parallel computing is all about compute performance. Here you present performance data obtained from e.g. running your newly developed algorithms and code on a parallel machine using some benchmark input data. Typically, you need to describe the parallel machine you used and the data that you used as input. The main results are usually performance graphs, typically speedup curves. You want to evaluate your code on different input data sets highlighting the strengths and weaknesses of your code. Don't just use best case scenarios. People will call you on that. Discuss the results obtained.

Figure 3 is a typical example of an experimental evaluation result. Such graphs are ususally created with GnuPlot.

## 6 Conclusions

You generally cover three things in the Conclusions section.

1. Conclusions

2. Summary of Contributions

3. Future Research

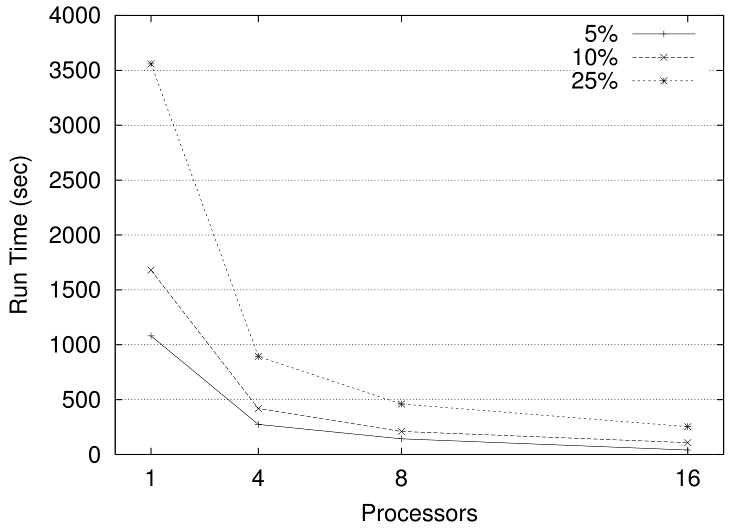

Figure 3: Measured Running Times Of Some Unknown Algorithm Implementation

Conclusions are not a rambling summary of the thesis: they are short, concise statements of the inferences that you have made because of your work. All conclusions should be directly related to the research question.

The Summary of Contributions will be much sought and carefully read by the readers. Here you list the contributions of new knowledge that your paper makes. Of course, the paper itself must substantiate any claims made here. There is often some overlap with the Conclusions, but that's okay.

The Future Research should indicate interesting new problems arising from your work. No paper ever solves everything. In fact, the best research papers lead to new research questions for other researchers to work on.

## References

[1] A. Baumann, P. Barham, PE. Dagand, T. Harris, R. Isaacs, S. Peter, T. Roscoe, A. Schüpbach, and A. Singhania. The multikernel: a new os architecture for scalable multicore systems. In *SOSP '09: Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, pages 29–44. Association for Computing Machinery, 2009.

[2] R. Brightwell, R. Riesen, K. Underwood, T. B. Hudson, P. Bridges, and A. B. Maccabe. A performance comparison of linux and a lightweight kernel. In *IEEE International Conference on Cluster Computing (CLUSTER'03)*, pages 251– 258. IEEE Computer Society, 2004.

[3] B. Gerofi, R. Riesen, M. Takagi, T. Boku, K. Nakajima, Y. Ishikawa, and R. W. Wisniewski. Performance and scalability of lightweight multi-kernel based operating systems. In *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 116–125. IEEE Computer Society, 2018.

[4] B. Gerofi, A. Santogidis, D. Martinet, and Y. Ishikawa. Picodriver: fast-path device drivers for multi-kernel operating systems. In *27th International Symposium on High-*

*Performance Parallel and Distributed Computing (HPDC '18)*, pages 2–13. Association for Computing Machinery, 2018.

[5] J. Ouyang, B. Kocoloski, J. Lange, and K. Pedretti. Achieving performance isolation with lightweight co-kernels. In *24th International Symposium on High-Performance Parallel and Distributed Computing*, pages 149–160. Association for Computing Machinery, 2015.

[6] R.Riesen, A. Barney Maccabe, B. Gerofi, DN. Lombard, JJ. Lange, K. Pedretti, K. Ferreira, M. Lang, P. Keppel, RW. Wisniewski, R. Brightwell, T. Inglett, Y. Park, and Yutaka Ishikawa. What is a lightweight kernel? In *5th International Workshop on Runtime and Operating Systems for Supercomputers (ROSS '15)*, pages 1–9. Association for Computing Machinery, 2015.