Energy-Efficient and High-Performance Parallel Video Decoding Techniques

by

Damla Kılıçarslan

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science
in
Electrical and Computer Engineering

Koç University

May 2012

Koc University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Damla Kılıçarslan

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____

A. Murat Tekalp, Ph. D. (Advisor)

_____

Serdar Taşıran, Ph. D.

_____

Çağatay Başdoğan, Ph. D.

Date _____

ABSTRACT

Emergence of high quality media applications results in larger data sizes as well as higher bitrates of digital multimedia contents, and their significant share on the overall Internet traffic. These lead to an increase in the energy consumption rates and performance requirements for real-time video decoding. In this thesis, we propose parallel video decoding solutions to provide real-time decoding performance with reduced energy consumption on multi-core devices. Various approaches of parallelism at data and task levels can be incorporated in video decoders, bringing efficiency in energy consumption rates and/or performance. We offer and develop several approaches for the H.264 standard including coarser-grained frame level and finer-grained macroblock level parallelism approaches. The implementations were conducted on a shared memory multi-core platform as an all software solution for real-time scalable video decoding. Faster real-time decoding performance with reduced energy consumption on multi-core processors is achieved. As future areas of research, further parallelization methods such as parallelism at slice level, and parallel decoding of consecutive groups of pictures on the H.264/SVC decoder are discussed.

ÖZET

Yüksek çözünürlüklü video sistemlerinin yaygınlaşmasıyla sayısal çoklu ortam içerikleri, daha büyük veri boyutları ve daha yüksek bit hızları gerektirmektedir. Bu durum çoklu ortam oynatıcılarındaki güç tüketim oranlarının ve gerçek-zamanlı görüntü çözümleme ihtiyacının artmasına sebep olmuştur. Bu yüksek lisans tezinde, çok çekirdekli cihazlarda düşük güç tüketimi sağlamak amacıyla gerçek-zamanlı paralel görüntü çözümleme yöntemleri önerilmektedir. Veri veya iş seviyesinde uygulanabilecek paralelleştirme algoritmaları sayesinde video çözümlemedeki enerji verimliliğinin ve performansın arttırılması mümkündür. Bu çalışmada paylaşımlı bellekte, çok çekirdekli bir platform üzerinde koşan yazılım çözümlerinde daha hızlı gerçek-zamanlı çözümleme ve düşük enerji tüketimi elde edilmiştir. İleriye dönük araştırma konuları olarak slice seviyesinde ve ardışık görüntü grupları (GOP) seviyesinde paralel görüntü çözümleme yöntemleri de ele alınmıştır.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my advisor Prof. Murat Tekalp, Dean of College of Engineering for his support and guidance during my M.Sc. study and research. He has inspired me in many ways not only by helping me ground my research efforts more firmly but also teaching me how to be passionate, focused and result-oriented in what I do. I wish to express my sincere acknowledgement to Prof. Öznur Özkasap for her guidance and advice throughout my studies. I am thankful for both of their contributions in many different ways including ideas, time and funding to make my M.Sc. experience more productive. I would also like to acknowledge Göktuğ Gürler, teaching and research assistant at the Multimedia, Vision and Graphics Laboratory, for his ideas and advice at the earlier stages of my thesis work.

I take this opportunity to acknowledge the funding sources that made my M.Sc. work possible. I was funded by The Scientific & Technological Research Council of Turkey (TÜBİTAK).

I am also greatly indebted to my employer Türk Telekom and my managers at Türk Telekom Group R&D Directorate for supporting me as well as for giving me the valuable time off from work to complete my studies. Without their understanding guidance and the innovative environment offered at the workplace it would have been impossible to accomplish this task.

Last but not least, I would like to thank my family; Emra Kılıçarslan and Adnan Kılıçarslan for all their love, encouragement and patience from day one. To them, I dedicate this thesis.

TABLE OF CONTENTS

LIST OF FIGURES

NOMENCLATURE

AVC	Advanced Video Coding
CMP	Chip Multi-Processors
GOP	Group of Pictures
HD	High definition
MB	Macroblock
MPEG	Moving Picture Experts Group
MPI	Message Passing Interface
NAL	Network Abstraction Layer
P2P	Peer-to-peer
RBSP	Raw Byte Sequence Payload
SVC	Scalable Video Coding
TLP	Thread Level Parallelism
VCL	Video Coding Layer
VOD	Video on demand

INTRODUCTION

1.1 Background and Motivation on Parallel Video Decoding

Human beings are wired to take in the visual and dynamic content in their environment. Our brains, anatomy and inborn sensory perception capabilities make us most easily drawn to visible action. Video and similar multimedia applications consequently prove to be one of the most effective, fast and pervasive ways to convey a message as part of the communicative media environments we are exposed to everyday. The ubiquitous presence of the Internet and digital content storage having defined the new age of connectivity led to the rise of network streamed video and multimedia content. The sum of all forms of video (TV, video on demand (VOD), Internet and Peer-to-Peer (P2P)) is expected to exceed 90 percent of global consumer traffic by 2014 [1]. With the emergence of high-definition (HD) and multi-view video formats the quality and definitions of video contents are getting more and more advanced as well. All these factors and the increasing end-user expectations on multimedia technologies constitute higher performance and energy demands and bring complexities to video codecs.

The increasing functionalities of embedded systems require higher power consumptions. Similarly, increasing amount of information sent over network communications results in enhanced compression algorithms to be applied to minimize the amount of data transmitted. Enhanced video compression algorithms require more power and computation complexity which in turn affects the overall performance of embedded systems and thus end-user experience on mobile devices. In conjunction to this, chip makers are constantly introducing multi-core chips for servers, desktops, laptops, post PC devices and smartphones. Current transistor technology limits the power performance of

single-core processors in accordance with Moore's law [2]. It is almost inevitable that parallel computing will be dominant in most machines and models with the rising trend of multi-core architectures. Every 3 years, the number of cores in chip multi-processors (CMPs) is expected to be doubled [3]. Applying parallel coding techniques on video codecs is only one aspect that comes along with the rising multi-threaded programming trend. With the increasing complexities in video formats there is a great demand for better performing, faster and less energy-demanding video encoders and decoders. Parallel computing proves to be one of the most effective solutions in many ways. In addition to providing extensive usage of processing power it also brings power saving and a smoother video playback experience as well.

With the emergence of mobile technology platforms and devices battery life is becoming a more crucial problem for designers. Applications that consume the least battery on mobile devices are more preferable to others and the fact that playing videos consume a significant amount of battery life on these devices makes it more critical to address this problem.

1.2 Contributions of This Thesis

This thesis presents the various methods for achieving better performing, energy-efficient video decoding by making use of multi-threaded architectures. Major contributions involve analyzing the energy efficiency of different parallel decoding algorithms. Existing parallel decoding algorithms either do not focus on the energy efficiency of the proposed method or is not an all-software solution that runs on a multi-core platform for the H.264/SVC video decoder.

The research presented in this thesis was published in the conference proceedings of ACM 2nd International Conference on Energy-Efficient Computing and Networking (E-

Energy 2011) which took place in Columbia University, New York between May 31 - June 1, 2011 [4].

Chapter 1 introduces the background and motivation for this research and main contributions of this thesis. Chapter 2 briefly summarizes the H.264 video decoding standard, reviews current literature on video coding and multi-threaded architectures and explains the overall structure and standards of a video sequence. It describes the general video coding models that have been standardized including MPEG-4, AVC/H.264 and Scalable Video Decoding (SVC) with key encoder/decoder parameters. Chapter 3 is sectioned into parts which propose possible parallel decoding methods. Chapter 4 first explains the testbed settings and benchmarks that are used to test and analyze the performance and energy-efficiency of the parallel decoding methods that are developed as part of this research. It then provides the test results for each algorithm and compares them with the original sequential decoding algorithm and with prior work done in the field. Finally, Chapter 5 draws conclusions out of the work accomplished in the dissertation and poses upcoming problems to be explained as part of future work.

LITERATURE REVIEW

Prior research conducted on parallel video codecs mainly focus on the performance of the decoder rather than its energy efficiency. The speedup, overhead and latency criteria were inspected more than the overall power consumption rate of the device running it. Part of the prior research carried out in this topic focuses on H.264/AVC standard whereas part of it is only MPEG-2 compatible. Following is a brief summary of the related literature survey.

2.1 The H.264 Video Decoding Standard

H.264/AVC is the latest video coding standard that can achieve flexibility and interoperability among different application areas of video transmission. It covers two layers; Video Coding Later (VCL), which is designed to efficiently represent video content and Network Abstraction Layer (NAL), which formats the VCL representation by adding appropriate headers so that it can be transmitted easily over various transport layers [5].

The encoder processes an input frame in units of macroblock (rectangular picture area of 16x16 samples of luma component and 8x8 samples of each of the chroma components) where each macroblock can be encoded in inter or intra mode. Within a macroblock, each block (subdivisions of a macroblock) is predicted from spatially neighboring samples. The prediction is subtracted from current macroblock and the residual block is transformed and quantized to give entropy encoded coefficients.

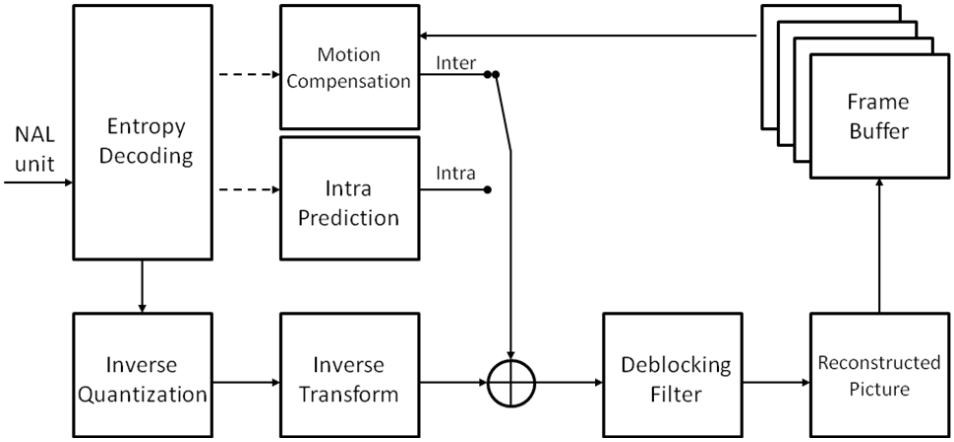


Figure 1 Decoder diagram of H.264

The decoder uses the header information of NAL units to determine the type of the payload (Raw Byte Sequence Payload – RBSP). It then does the entropy decoding to produce a set of quantized coefficients which are then scaled and inverse transformed. An in-loop deblocking filter within the motion compensation helps reduce the visual artifacts as shown in Figure 1.

The decomposition hierarchy of a video stream in H.264 is arranged in 6 main layers in the following order: Video Sequence, Group of Pictures (GOP), Picture, Slice, Macroblock and Block.

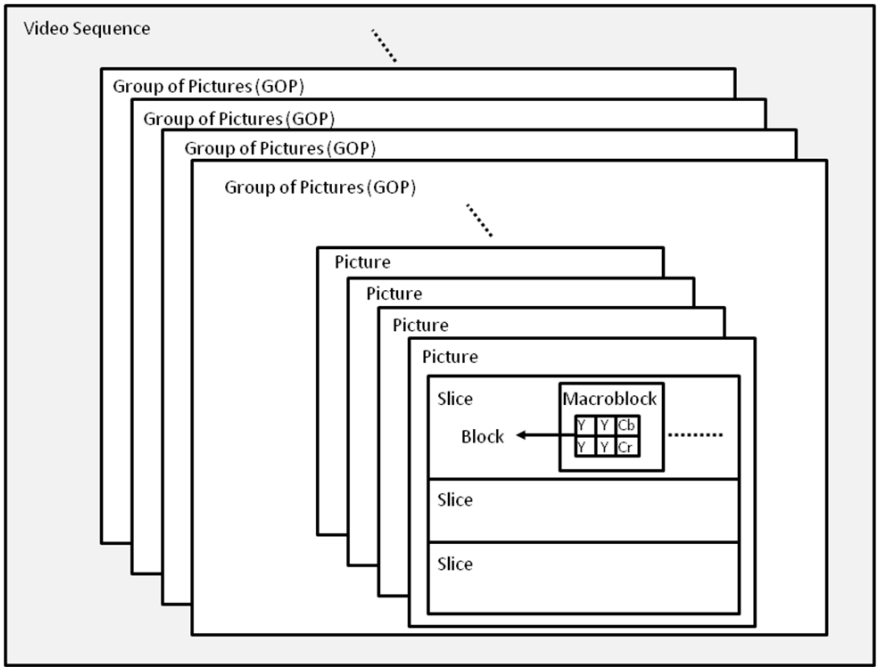


Figure 2 Hierarchical organization of H.264 video stream decomposition

2.2 Related Work on Multi-Threaded Video Encoding

One of the noticeable studies that have been carried out on H.264 standard focuses on the encoder side rather than the decoder side. A coarser grain implementation carried out at slice, GOP and frame levels for the H.264 encoder was presented by Rodriguez et al. in [6]. They propose a hierarchical parallelization of H.264 encoders well suited to low cost clusters using MPI message passing parallelization at GOP and frame levels. They show that GOP level parallelism can provide a good speed-up but comes at a cost of relatively high latency. The slice level parallelism on the other hand gets less efficiency but with lower latency. An optimization compromising between speedup and latency by combining both approaches is formed.

2.3 Related Work on Multi-Threaded Video Decoding

One of the novice research areas on the decoder side was carried out in [7]. It exploits both the coarse grained parallelism approach at GOP level and the fine grained parallelism within the pictures at slice level on the MPEG-2 decoder. In this work, a comparative evaluation of methods is provided and very good speedups and locality properties are demonstrated.

A similar GOP level parallelization without a start-code scanner for the H.264 decoder is proposed in [8] GOP-level parallelism provides high scalability but requires more memory resources. The threads running in parallel to decode separate GOPs do not have to wait for each other since this proposal assumes a closed GOP structure meaning there aren't any references between consecutive GOPs. The tests on a cluster of 5 machines each with 2 processors and 4 cores show a linear speedup if there is no memory shortage. When all the processes run on the cores of the same machine the speed up can reach up to around 2.5 on 8 cores. This is due to the effect of cash population on a shared memory device.

For the H.264 decoder finer grain approaches at MB level were examined in [9], [10] and [11]. [9] Jikes et al. point out the inherently sequential nature of the control intensive front end of the H.264 decoder and that preparsing could be a functional parallelization technique to resolve this bottleneck. They propose two novel methods to enhance the efficiency of this technique: (i) a custom preparsing technique to resolve control dependencies in the input stream and expose MB level data parallelism, (ii) an MB level scheduling technique to allocate and load balance MB rendering tasks. They managed to achieve up to 60% speedup over dynamic scheduling and up to 15% speedup over static compile time scheduling for more than four processors.

In [10], Mesa et al. bring a closer perspective on the scalability of the MB level parallel decoding techniques. They present a quantitative analysis of the main bottlenecks of the application and estimate the acceleration levels that are required to make the MB-level parallelism scalable. The strategy involves three steps: (i) creating a model for predicting the maximum performance that can be obtained taking into account the variable processing times and thread synchronization overhead, (ii) implementing the model on a real multi-processor machine including a comparison of different scheduling strategies and a profiling analysis for identifying the performance bottlenecks and (iii) identifying the performance driven bottlenecks by making use of a trace-driven simulation methodology.

Another study that specializes in the scalability of MB-level parallel decoding is [11]. In this study Meenderinck et al. propose a novel strategy, called 3D-Wave, which is mainly based on the observation that inter-frame dependencies have a limited spatial range, which allows certain MBs of consecutive frames to be decoded in parallel.

In [12] authors propose a way for better efficient coordination of parallel threads in H.264/AVC decoder. The experimental results shows that the H.264/AVC decoder proposed parallelization technique achieves 25% speedup compared to existing parallelization techniques.

Baaklinni et.al. explore the natural existence of parallelism in H.264 decoder software without modifying the encoder part. They propose a way for decoding the luminance and chrominance signals in parallel. The results indicate that using 2 cores to decode the luma and the chroma signals in parallel gives a gain of 15-20% of the decoding processing time and their combination in a functional pipeline over four cores or more can result in a gain of 60% compared to the original sequential execution [13].

Two new approaches are proposed by Samsung Electronics; software memory throttling and fair load balancing. Software memory throttling limits the number of cores involved in the parallel motion compensation to achieve power-saving and better speedup.

The fair load balancing is applied on deblocking filter reduces the load imbalance due to the original static partitioning method. This allows up to 24% speedup on two different symmetric multicore platforms[14].

In the paper “Adaptive multithreaded H.264/AVC decoding”, Richter et. al. examine two variants of multithreaded video decoding with distributed synchronization. First one is optimized for minimum latency decoding and the latter maximizes the throughput at the cost of higher latency. Experimental results demonstrate scaling abilities of up to factor 3.5 on a quad-core machine and show that a 4k resolution decoding is feasible in real-time on a mid-range PC hardware of that time (2009) [15].

2.4 Related Work on Low-Power Video Decoding

A study carried out by Soner Yaldiz, Alper Demir and Serdar Tasiran make use of stochastic modeling and optimization for energy management in multi-core systems[16]. In this study they capture spatial and temporal correlations among work load tasks and use them in novel mathematical formulations to obtain energy efficiency. By making use of dynamic voltage scaling, this method is applied on MPEG-2 video decoding and experimental results show significant energy savings.

Contrary to the many all-software approaches listed above, there has been a hardware based implementation that focuses on the power efficiency of the H.264 decoder as well. An application-specific integrated circuit architecture was presented in [17]. In his Ph.D. dissertation Finchelstein implemented several architecture optimizations that reduce the system power of a high-definition video decoder.

MULTI-THREADED VIDEO DECODING ALGORITHMS

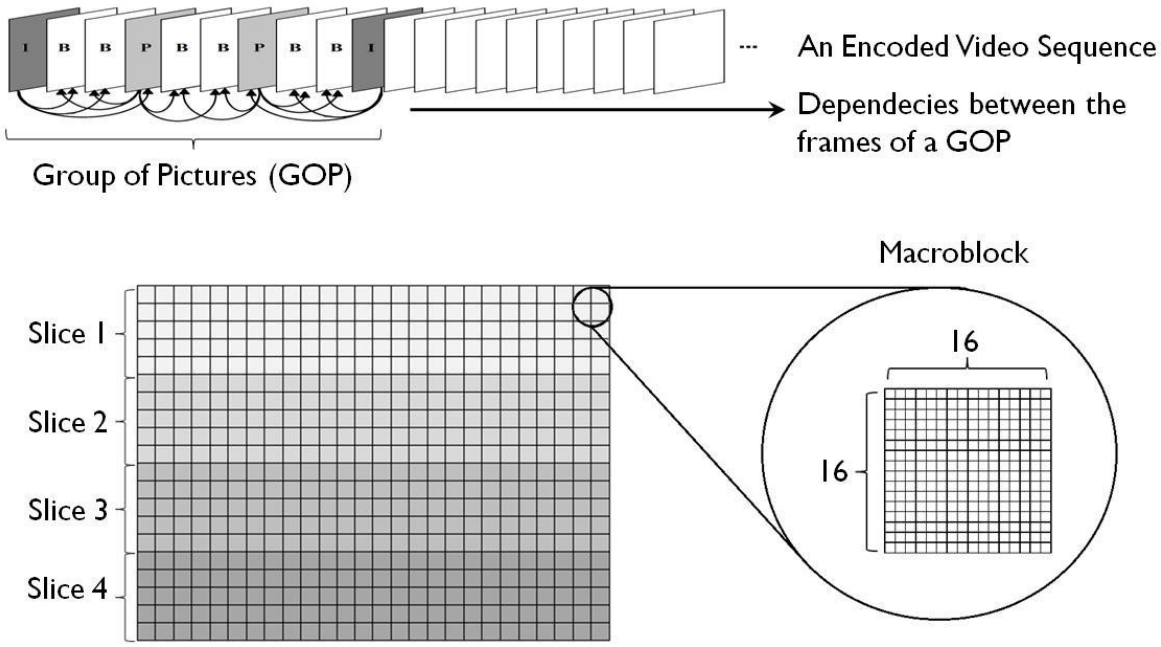


Figure 3 Hierarchical structure of an encoded video sequence

The ubiquity of super high resolution and 3D video content demand high computational power in video decoders. Task-level parallelism and data-level parallelism offer solutions to this problem in different ways. Although task-level parallelism has reached to saturation, data-level parallelism offers many different granularity levels. An encoded video structure is composed of a hierarchical structure which includes independently decodable parts namely; GOP, frame, slice, MB and block levels. Each level has its own challenges and advantages.

3.1 Macroblock Level Parallelism

H.264/AVC performs block-based video coding approach in which frames are partitioned into rectangular areas, known as macroblocks (MB). The size of a MB is 16x16 pixels for luma layer and 8x8 for chroma layers for source sequences in 4:2:0 YUV format. A MB is either spatially or temporally predicted depending on the type of the frame [5]. MBs of predictive (P) or bi-direction predictive (B) frames can be both spatially or temporally predicted whereas the prediction for MBs in intra coded (I) frames is restricted to spatial prediction.

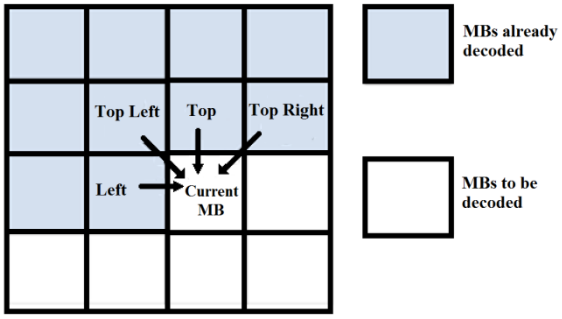
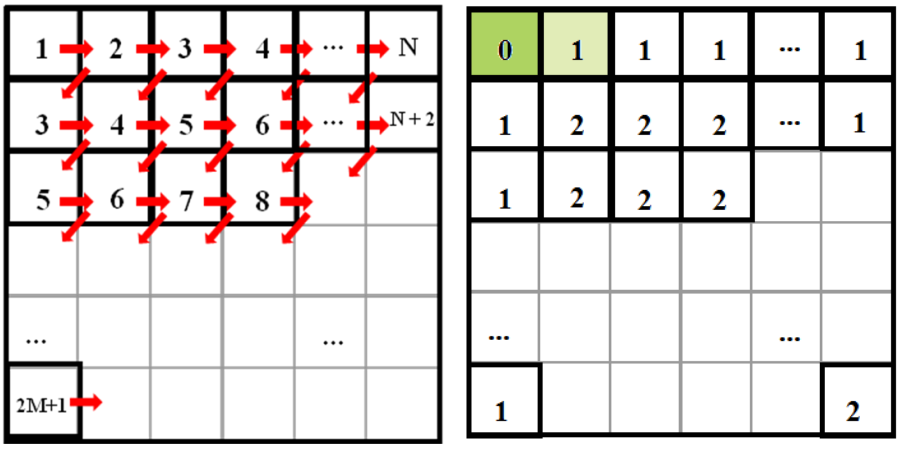


Figure 4 Spatial dependencies between neighboring MBs

A certain decoding order was applied for spatially predicted MBs, as depicted in Figure 4. Since encoding is performed in raster scan order, MBs can be decoded in the same order. However, it is possible to decode the MBs in a different order as long as all the dependent MBs are decoded prior to the current MB. Note that if MBs are temporally predicted, there are no such restrictions for the decoding order of the remaining MBs.



a) **b)**

Figure 5 For a frame with MxN (width x height in MB)
a) Decoding order for MBs and their successor(s)
b) Number of references for MBs

The dependency hierarchy enables decoding of multiple MBs simultaneously. One such possibility is depicted in Figure 3(a), revealing that if MBs with decoding order 1 and 2 are processed then two MBs (numbered as 3) can be decoded simultaneously. Note that the number of MBs that can be processed in parallel increases in the later stages of the decoding process.

In contrast to the sequence level parallelism (described in Section 3.3), macroblock level parallelism is a lot finer grain and requires considerable modifications on the decoder source code. The Intel Thread Building Blocks (TBB) library was chosen to implement the parallel algorithm [18]. Due to the dependencies shown in Figure 4 left, top-left, top and top-right MBs should be fully decoded before the current MB can be started. The decoding of MBs can be represented as a directed acyclic graph with each node of the graph representing the corresponding decoding of that MB by one processor.

Decoding of each MB is considered as a task and the numbers in Figure 5(b) represents the number of references required to start processing that task. Consequently, the upper left most MB can be initiated as the first MB. Since it is the only MB with no

requirements, its reference count is set to zero. Upon completion of a task, the reference count of the successor MB(s), which are represented with the arrows in Figure 5(a), is decremented. Thus, once the first MB is decoded, the second task (the MB next to it) becomes available. Likewise, upon completion of the second MB, reference count of two successor MBs is decremented making them available parallel decoding. The process continues until all MBs are decoded in that frame.

3.2 Slice Level Parallelism

Macroblocks of a picture are organized in slices, each of which can be parsed independently of other slices in a picture. This approach makes use of that arcthiecture to offer parallel decoding of the slices of a picture.

The implementation was carried out by modifying the Open SVC H.264/AVC decoder by using the Boost C++ Parallel Libraries.

3.3 Sequence Level Parallelism

In this approach, the YUV format of the original video sequence is split into n threads. This splitting pattern allows load balancing over threads that perform the decoding process later on. Each individual split YUV sequence is encoded using the MPEG-4, AVC/H.264 or SVC standard and decoded separately on different cores simultaneously as depicted in Figure 6.

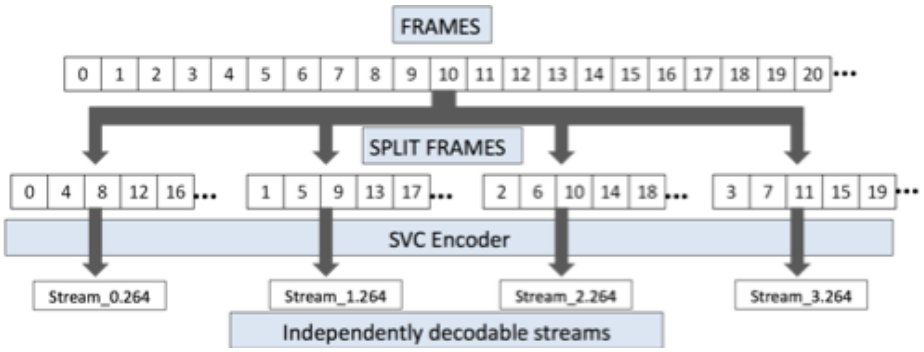


Figure 6 Splitting the original video in multiple threads to be decoded individually with four threads

During the decoding process, a size 30 GOP buffer is used. Each independently decodable stream is merged into one video sequence inside the frame buffer during the course of decoding. When the buffer is full, the threads are put to sleep for one second since there is no need to decode more frames until the previous ones are played. This method allows the decoding process to slow down whenever it goes over the speed of the actual display rate. Therefore, it avoids unnecessary CPU usage allowing further decrease in the overall energy consumption. This approach trades off increased parallelism with the encoding efficiency because consecutive frames are distributed over different cores. On the other hand, if frame level parallelism is ordered according to the levels of hierarchy in hierarchical B-pictures encoding, then the encoding efficiency will not be affected at the cost of slightly reduced parallelism.

ENERGY-EFFICIENCY AND PERFORMANCE MEASUREMENT RESULTS

The performance and energy consumption measurement tests were carried out on a laptop running on Intel® CoreTM i7 720-QM quad-core processor at 1.60 GHz with 6M cache. The Enhanced Intel Speedstep® Technology (EIST), Turbo Boost Technology and the Intel® Hyper-Threading Technology offered with this processor provide the adjustability on the processor performance to observe the changes in the energy consumption for a given task.

Tests were conducted on input videos Iceberg (video with still background and moving camera), Race (video with fast-moving objects and moving camera) and Rena (video with still camera and background with moving figure) for the frame level and MB level parallelism approaches.

In order to measure the overall energy consumption of the whole device, its instantaneous outlet power consumption was measured over the time span while the decoding takes place by making use of a commercially available power meter called WattsUp PRO power meter [19]. The timing measurements were carried out by the tick_count class of the Intel TBB library.

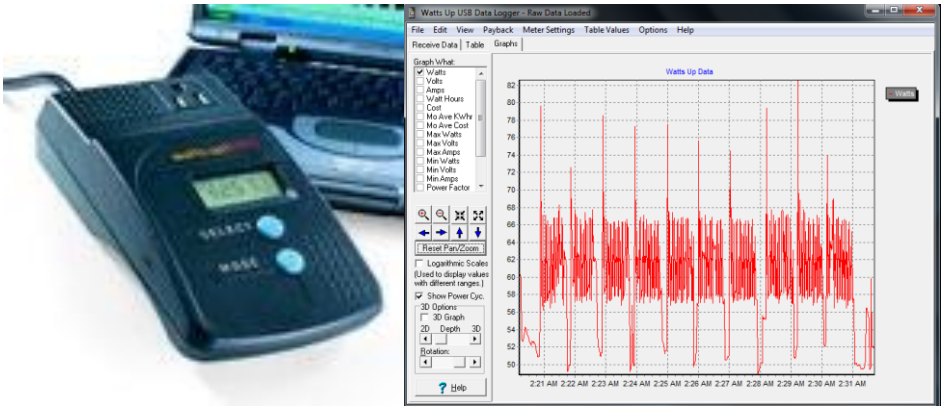


Figure 7 Power Meter used to measure Instantaneous Outlet Power Consumption (WattsUp? PRO Meter) and a screenshot of its user interface

4.1 Macroblock Level Parallelism Measurements

The three input videos Iceberg, Race and Rena were encoded using the SVC extension of the H.264/AVC encoder with base and enhancement layers. The frame rate was set to 30 fps and quantization parameters of level 0 and 1 were 46 and 34 respectively. Figure 8 shows the power consumption of the decoding process measured using a power meter when the player is switched off.

The macroblock level parallelism approach offered a relatively smaller energy efficiency difference compared to frame level parallelism since the overall effect of the region parallelized in decoding of MBs (I-frames only) had a minor impact on the overall decoding performance. When compared to previous work carried out in MB level parallel decoding methods the speedups are consistent with the static scheduling average speedup results up to 8 cores presented in [9] and [10]. This resulted in a smaller improvement on percentage changes of the energy consumption rates of the decoder.

The timing measurement sets carried out in this section observe the performance change introduced with parallel MB decoding. Speedups calculated with respect to the original Open SVC decoder running of sequential MB decoding algorithm for 2, 4 and 8 threads are shown in Figure 10.

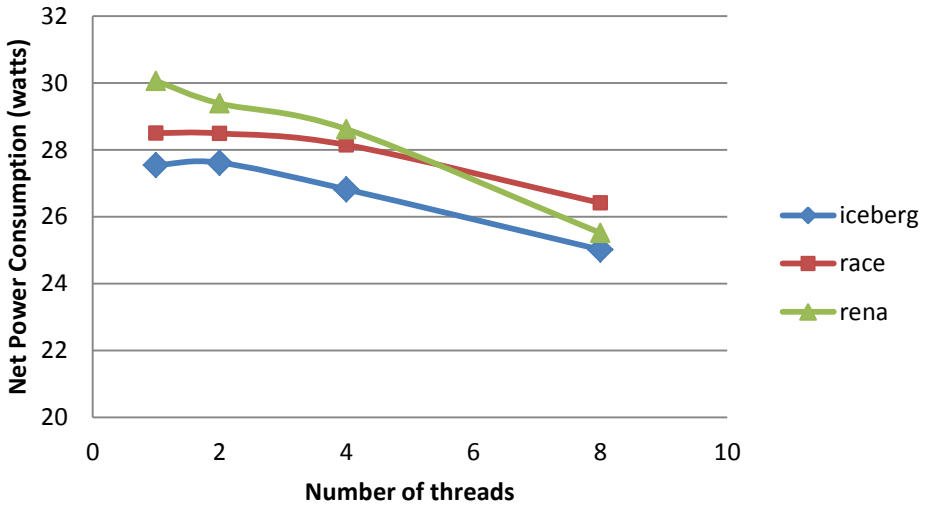


Figure 8 Net average power consumption rates in MB level parallelism

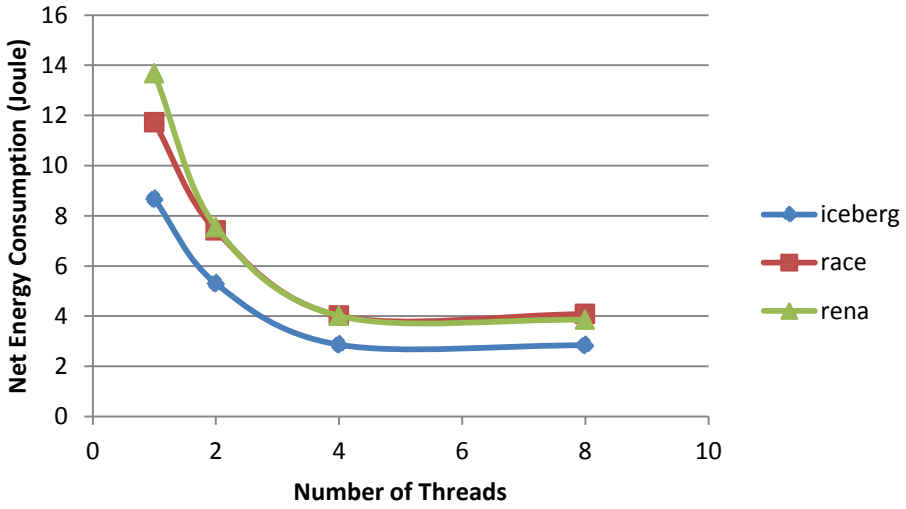


Figure 9 Net energy consumption amounts in MB level parallelism for decoding of I-frames only

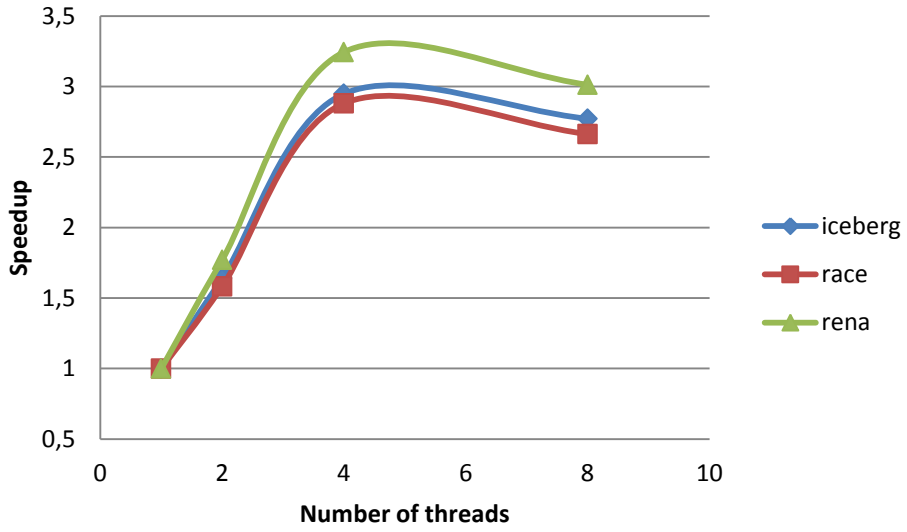


Figure 10 Speedups achieved in MB level parallelism for decoding of I-frames only

4.2 Sequence Level Parallelism Measurements

The three input videos Iceberg, Race and Rena were encoded from n split video sequences using the MVC mode with a quantization parameter of 22 and frame rate of 30 fps. The power consumption rate measurements were carried out by a power meter over the course of the whole decoding process as the decoded frames were being played at the same time.

The average idle power consumption rate of the laptop was taken to be 30 watts throughout the measurements. Figure 11 shows net power consumption rates per number of threads that are calculated by subtracting the idle power from the average instantaneous power consumption rates throughout the decoding process.

This approach trades off increased parallelism with the encoding efficiency since consecutive frames of the original video sequence need to be distributed over different cores. However, this approach offers better load balancing among threads. Since similar frames are decoded over different cores, each core gets a balanced amount of tasks bringing

a more efficient parallelism approach. The energy measurement results indicate that with 8 threads 20% power efficiency can be achieved using this technique and elapsed times for the complete decoding process decrease considerably.

To observe the true effect of parallelism, the elapsed times included only the time span for the decoding portion of the whole process excluding the frame buffer storage time and display times. Speedups computed for decoder running on 2, 4 and 8 threads are shown in Figure 13.

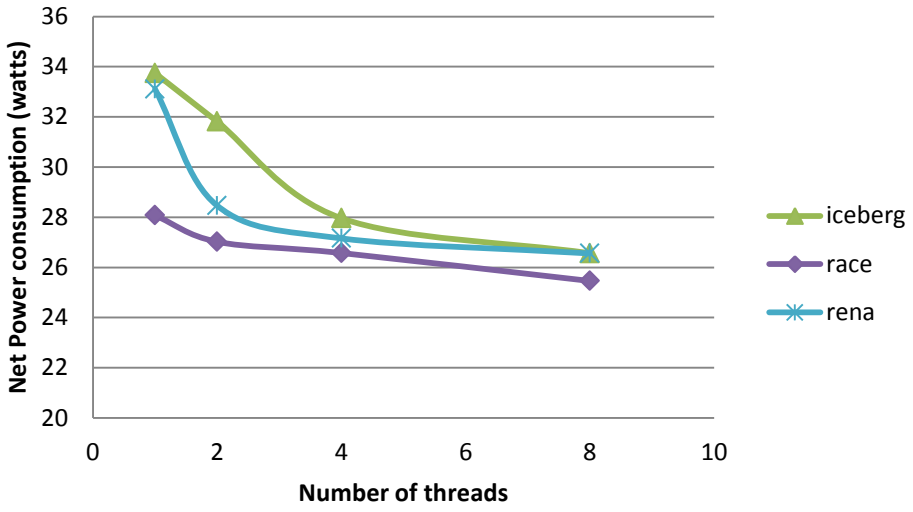


Figure 11 Net average power consumption rates in sequence level parallelism

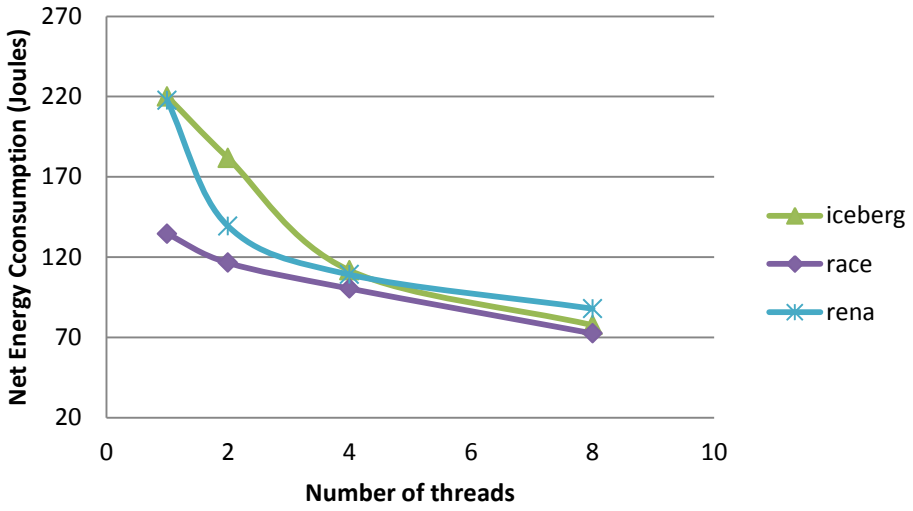


Figure 10 Net energy consumption amounts in sequence level parallelism

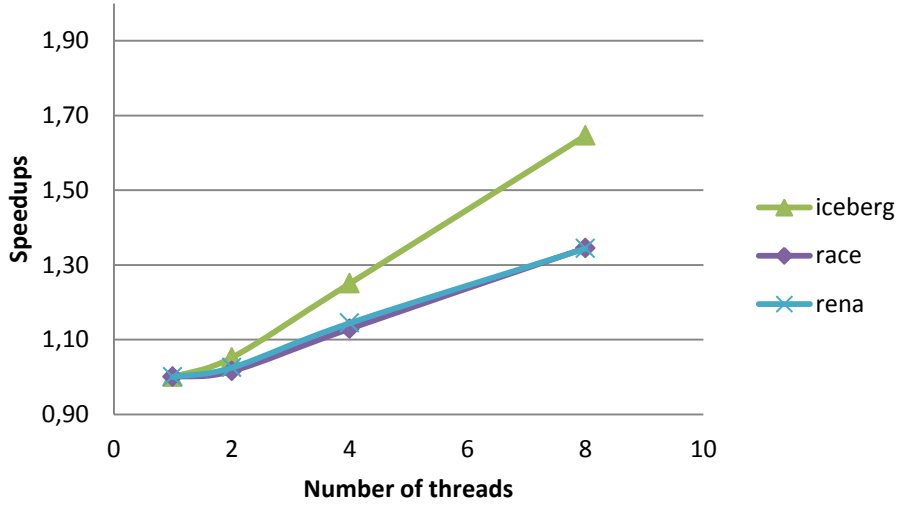


Figure 13 Speedups achieved in sequence level parallelism

CONCLUSION

5.1 Conclusion

Frame level and macroblock level parallelization techniques described in this work are at the two extremes of granularity scale of parallelism for video decoding. The frame level approach is based more on data-level parallelism, which is coarse-grained, and relatively simpler compared to other task-based parallelism techniques. On the other hand, the macroblock level parallelism approach is a lot finer grain and thus requires more complicated algorithms to achieve similar or better performance speed-up and energy savings.

In order to make an accurate comparison of both techniques the decoding of B-frames have to be parallelized in addition to the parallelization of I-frames in macroblock level parallelization approach. Only then graphs 8, 9, 10 and 11, 12, 13 will be truly comparable. However since the decoding of I-frames are the most computationally demanding portion of the video sequence the results are still effective enough to outline a few major differences between the two approaches. One of these differences is that speedups in macroblock level are more likely to reach saturation faster than in frame level approach as the number of threads increases. This is due the overhead introduced by parallelization at such finer-grained levels like macroblocks.

Parallelization of video decoders at the software level requires careful synchronization between the dependent tasks in a video decoder. Most of the time, high level implementations are not sufficient and major modifications on the original video decoder source code are required. The key strategy for determining the regions to parallelize inside the decoder relies on careful assessment of the function declarations and the execution flow of the program.

A careful optimization between the choice of level at which parallelism will be applied and the overhead caused after it, can lead to great impacts both in terms of performance and energy efficiency in video decoders. Since video applications have become a major part of the global Internet traffic, developing better performing and more energy efficient video decoders will lead to a more fulfilling playback experience at the user-end, longer battery life sustained on mobile devices and, most importantly, massive amounts of energy saving on the global video traffic per user each year.

5.2 Future Areas of Research

Further enhancements on the macroblock level of parallelism may include parallelizing B-frames' macroblock decoding functionality. Since B- frames are not intra-predictable frames, the dependencies of MBs on other frames will be more challenging than parallel MB decoding on I-frames only. This feature will surely bring additional performance and energy efficiency when implemented with low overhead and careful load balancing. Moving on to higher levels from the macroblock decoding region, slice level parallelism and GOP level parallelism are other intriguing areas of research multi-threaded video decoding techniques.

Scalability of slice level approach is limited since the number of slices is decided by the encoder. Additionally, increased number of slices within a frame increases the bit rate as well. GOP level on the other hand do not require synchronization between threads since GOPs are entirely independent. This approach may offer higher scalability however it requires considerable amount of memory and thus may result in latency during real-time play back.

BIBLIOGRAPHY

[1] Cisco Visual Networking Index: Forecast and Methodology, 06 02, 2010. http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html

[2] D. Geer, Industry trends: Chip makers Turn to Multicore Processors, Computer, vol. 38, no. 5 (2005), 11-13.

[3] P. Stenstrom, Chip-multiprocessing and beyond, The Twelfth International Symposium on High-Performance Computer Architecture (2006), 109.

[4] D. Kılıçarslan, C. G. Gürler, Ö. Özkasap, A. M. Tekalp, Energy Efficient Video Decoding on Multicore Devices, ACM 2nd International Conference on Energy-Efficient Computing and Networking (e-Energy) (2011).

[5] T. Wiegand, G. J. Sullivan, G. Bjontegaard and A. Luthra, Overview of the H.264 / AVC Video Coding Standard, IEEE Transactions on Circuits and Systems for Video Technology, (2003), 560-576.

[6] A. Rodriguez, A. Gonzalez, M. P. Malumbres, Hierarchical Parallelization of an H.264/AVC Video Encoder Parallel Computing in Electrical Engineering, PAR ELEC, (2006), 363-368.

[7] A. Bilas, J. Fritts, J. Pal, and S. Paper, Real-Time Parallel MPEG-2 Decoding in Software, 11th International Parallel Processing Symposium (IPPS) (1997).

[8] A. Gurhanli, C. C. P. Chen, S. H. Hung; Grad. Inst. of Electron. Eng., Nat. Taiwan Univ., Taipei, Taiwan Signal Processing Systems (ICSPS), (2010), 627-630.

[9] C. Jike, N. Satish, B. Catanzaro, K. Ravindran, K. Keutzer, Efficient Parallelization of H.264 Decoding with Macro Block Level Scheduling, IEEE International Conference on Multimedia and Expo, (2007), 1874-1877.

[10] M. Mesa, A. Ramirez, X. Martorell, E. Ayguade and M. Valero, Scalability of Macroblock-level Parallelism for H.264 Decoding, ACACES. (2008).

[11] C.H. Meenderinck, A. Azevedo, M. Alvarez, B.H.H. Juurlink, A.Ramirez. Parallel Scalability of H.264, Proceedings of the first Workshop on Programmability Issues for Multi-Core Computers, Geteborg, Sweden (2008).

[12] S. H. Jo, S. Jo and Y. H. Song, Efficient Coordination of Parallel Threads of H.264/AVC Decoder for Performance Improvement (2010).

[13] E. Baaklini, H. Sbeity, S. Niar, and N. Amaneddine, H.264 Color Components Video Decoding Parallelization on Multi-core Processors. In Proceedings of the 2010 13th Euromicro Conference on Digital System Design: Architectures, Methods and Tools (DSD '10). IEEE Computer Society, Washington, DC, USA (2010), 785-790.

[14] K. H. Sihn, H. Baik, J.T. Kim, S. Bae and H. J. Song, Software Lab., Samsung Electron. Acoustics, Speech and Signal Processing, ICASSP (2009), 2017 – 2020.

[15] H. Richter, B. Stabernack, and E. Müller, Adaptive multithreaded H.264/AVC decoding, In Proceedings of the 43rd Asilomar conference on Signals, systems and computers IEEE Press, Piscataway, NJ, USA (2009), 886-890.

[16] S. Yaldiz, A. Demir and S. Tasiran, Stochastic Modeling and Optimization for Energy Management in Multi-Core Systems: A Video Decoding Case Study, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, (2008).

[17] D. F. Finchelstein, Low-Power Techniques for Video Decoding, Ph.D. Thesis, MIT, 2009.

[18] Intel ® Threading Building Blocks Tutorial
http://www.threadingbuildingblocks.org/documentation.php

[19] WattsUp Products.
https://www.wattsupmeters.com/secure/products.php?pn=0

DAMLA KILIÇARSLAN

Türk Telekom Ar-Ge, İTÜ Ayazağa ARI-4 Binası, Maslak İstanbul TURKEY
+90532 - 236 - 4814 • damlakilicarslan@gmail.com
Birthday: June 25th 1988

Experience

R&D Engineer, Türk Telekom – Istanbul, Turkey May 2011 – Present
▪ Assisting in the transfer of developed technology to business units including software, documentation and results within the University Collaboration Team
▪ Managing multiple R&D projects; reorient the projects based on business perspective

Intern, Ericsson – Ankara, Turkey June 2009
▪ Assistance for the sales and management department of the regional office

Intern, Southern California Edison – Santa Ana, CA June – Sep. 2008
▪ Assistance for Asset Management & System Reliability department

Intern, Likom Defense Mechanisms and Programming Co., Ankara, Turkey June – July 2007
▪ Database construction and technical report preparation for product development department

Education

Koç University, İstanbul, Turkey 2009 – 2011
▪ M.Sc. in Electrical and Computer Engineering / GPA: 3.67
▪ TUBITAK scholarship and Koç University M.Sc. Graduate scholarship
▪ Researcher at Multimedia Vision and Graphics Lab for Dean Professor Murat Tekalp
▪ Teaching Assistant for Signals and Systems, Advanced Programming, Network Security classes
▪ Publication: Kılıçarslan, D. Gürler, C. G., Özkasap, Ö., Tekalp, A. M. 2011. Energy Efficient Video Decoding on Multi-Core Devices. In Proceedings of the 2nd International Conference on Energy-Efficient Computing and Networking (e-Energy '11). ACM, New York, NY, USA.

Université Paris – Sorbonne Paris IV, Paris, France summer 2009
▪ French language courses (Cours de Civilisation Française de la Sorbonne Certificat)

Bilkent University, Ankara, Turkey
2005 – 2009
▪ B.Sc. in Electrical and Electronics Engineering / GPA: 3.47
▪ 80% Merit Scholarship - High Honor Student

University of California, Los Angeles 2007 – 2008
▪ Exchange Student in Electrical Engineering - Dean's Honor List /GPA: 3.48
▪ Lab assistant at Center of High Frequency Electronics
▪ Teaching Assistant for Engineering Physics class and summer researcher at Innovate Lab

Istituto Italiano di Cultura di Ankara, Ankara, Turkey 2006 – 2008

Europass, Florence, Italy and Cultura Italiana, Bologna, Italy summer 2006 & 2007
▪ Advanced level Italian language courses (Certificato di Italiano come Lingua Straniera)

TED Ankara College Private High School Foundation Ankara, Turkey 2002 – 2005
▪ Turkish Ministry of Education High School Diploma: 4.98/5.00
▪ International Baccalaureate Bilingual Diploma

Computer Skills

Programming languages: C, C++, Objective-C, Java, J2ME, MATLAB
Other: Microsoft Office, Linux OS, Mac OS, iOS, Adobe Photoshop, iPhone application development

Languages

English (bilingual), Turkish (native), Italian (fluent), French (intermediate)