Energy Efficiency of Out-of-Order CPUs: Comparative Study and Microarchitectural Hotspot Characterization of RISC-V Designs

Odysseas Chatzopoulos[†] George Papadimitriou[†] Wing Shek Wong[§] Dimitris Gizopoulos[†]
[†]*University of Athens, Greece {od.chatzopoulos | georgepap | dgizop}@di.uoa.gr*
[§]*Intel, Austin, Texas wing.shek.wong@intel.com*

Abstract

Building on the power of open-source RISC-V CPU designs at the register-transfer level (RTL) we evaluate the energy efficiency of the state-of-the-art open-source out-of-order (OoO) RISC-V microarchitecture (SonicBOOM) at three different design points of increasing aggressiveness. We measure the contributions of all major hardware structures and identify their power consumption. Our findings can assist microprocessor designers in making informed decisions regarding the microarchitectural trade-offs, enabling the development of more energy-efficient CPUs by focusing on the major power consumption contributors and their performance criticality.

1. Introduction

Microarchitecture plays a crucial role in determining the power consumption of a processor or a computer system [1]. Power consumption is a significant concern in modern computing devices due to limitations in battery life, energy efficiency requirements, and the need for thermal management [2]. By measuring and optimizing microarchitecture choices, designers can contribute to energy-efficient solutions that align with the growing demand for sustainable and power-conscious technologies [3–5]. With the growing popularity of the RISC-V instruction set architecture (ISA) and the large variety of different microarchitectures that implement the ISA, understanding the combined performance and power implications of such choices on RISC-V cores becomes increasingly vital. This paper builds on top of a publicly available OoO RISC-V CPU (SonicBOOM) and aims at fulfilling this goal. In this paper, (1) we present a comprehensive RTL power estimation flow, using a comprehensive, end-to-end combination of commercial and open-source tools, (2) we assess the impact of microarchitecture on power consumption, targeting three different configurations of SonicBOOM [6] (i.e., MediumBoom, LargeBoom, MegaBoom), (3) we evaluate and present insights on how major microarchitectural components contribute to the power consumption of the entire microprocessor chip, and (4) we present the performance per watt evaluations for ten benchmarks and each of the three BOOM configurations we used in this study.

2. Tools, Models, Frameworks

We employ a set of both open-source and commercial tools to evaluate the effects of microarchitecture on chip power consumption. These tools enable fast design space exploration of different microprocessor configurations, providing metrics such as performance, area, and energy efficiency. In this section, we enumerate and briefly describe these tools.

1) **Chipyard** is a widely-used, open-source framework that enables the design and evaluation of full-system hardware using an agile development approach [7].
2) **SonicBOOM** is the 3rd generation of BOOM, the state-of-the-art, open-source, out-of-order RISC-V core. It is written in Chisel and is highly parameterized, allowing for fast design-space exploration and optimization for a wide range of target applications [6, 8]. Figure 1 presents a microarchitectural diagram of the BOOM core.
3) **ASAP7** is a 7nm predictive process design kit (PDK). It was developed in collaboration with Arm for academic use. The kit is based on the assumptions for the 7nm technology node and is not tied to any specific foundry for manufacturing [9].

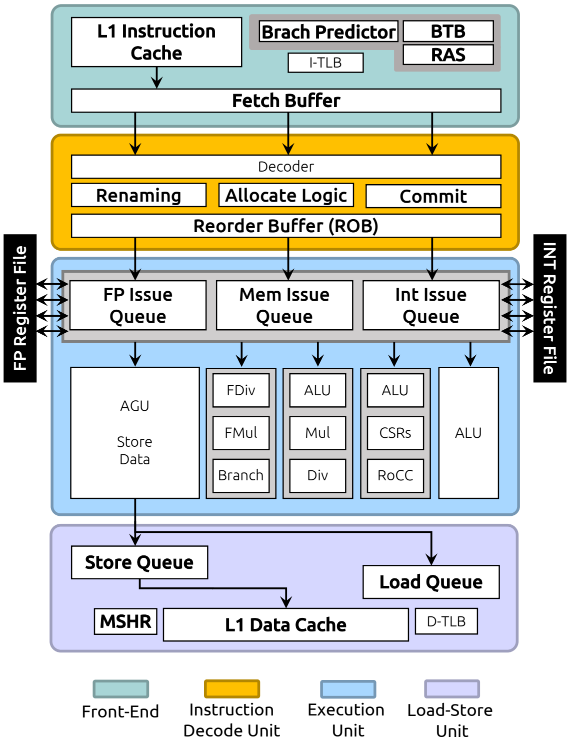

Figure 1: SonicBOOM microarchitecture. Different background colors show the distinct major units of the core.

4) **Verilator** is a high-performance simulator for the Verilog hardware description language (HDL) [10]. It is renowned for its speed, reputed as the fastest open-source simulator for Verilog HDL [11].

5) **Cadence Joules** is an RTL power estimation tool that achieves both high accuracy and relatively high speed [12]. Joules can evaluate all three parts of power in digital CMOS designs, i.e., **Leakage (Static) Power**, **Internal (Short Circuit) Power** and **Switching Power** [13].

3. Experimental Setup

This section describes the experimental methodology we follow to assess the impact of microarchitecture on chip power consumption. We describe the experimental flow explaining how we combined the use of the tools described in Section 2, as well as the benchmarks and the target BOOM configurations employed for our analysis.

3.1. Experimental Flow

Our workflow is illustrated in Figure 2. The first step is to select the target Chipyard SoC configurations ❶ that contain three different BOOM setups, i.e., MediumBOOM, LargeBOOM and MegaBOOM, as shown in Table 1. MediumBOOM is a 2-wide core, LargeBOOM is 3-wide and MegaBOOM is 4-wide. All three configurations implement the BOOM microarchitecture, as depicted in Figure 1. The microarchitectural components highlighted in bold within this figure indicate the components under consideration for our study. It is clear that we have encompassed the majority of the microarchitectural components.

Our target configurations define the entire chip including the BOOM core, caches, and peripheral that are included in Chipyard as customizable, Chisel based, generator IP blocks ❷. The Chisel-based designs undergo a build process, which initially converts them into FIRRTL intermediate representation. Subsequently, the intermediate representation is processed through a series of predefined transforms, such as memory macro mapping, resulting in the creation of the target Verilog ❸. This target Verilog is fed into Verilator that transforms it into a fast, multithreaded, cycle-accurate simulator ❺ that can produce cycle-by-cycle signal traces for the entire chip. The default Verilator configuration has two limitations, preventing us from tracking all signals in the design: (1) some signal names start with an underscore which Verilator by default ignores during trace file generation, and (2) some unpacked arrays are over the size limit that Verilator traces by default. We address these issues by passing the appropriate command line arguments to Verilator during the simulator build process ❹.

We utilize a diverse set of ten benchmarks, each consisting of an average of 1.4 million dynamic instructions (we have confirmed that this is a typical workload execution time for RTL simulations in industry setups). To ensure accurate results in terms of power and performance for all benchmarks, particularly those involving a significant number of memory accesses, we perform cache warm-up by executing each benchmark twice. *Towers*, *Software Multiply*, and *QSort* benchmarks are part of the RISC-V toolchain, while *Basicmath*, *Bitcount*, *Dijkstra*, and *FFT* benchmarks are part of the MiBench benchmark suite [14]. *Int_Add* and *Mem_Load* are custom benchmarks designed to stress the integer and memory pipelines, respectively, while the *Dhrystone* benchmark is a well-known, integer-focused benchmark widely used for comparing the performance of different cores.

Running the benchmarks on our fast multithreaded simulator, we produce 30 trace files ❻, each encompassing every signal in the respective design for every execution cycle. The trace files that Verilator produces are in the Value Change Dump (VCD) format. Each file varies in size, falling within the range of 30 to 70 gigabytes. While the VCD format is commonly used and well-accepted, we found the size of the files to be excessively large for practical handling and analysis. To overcome this challenge, we convert these files into the FSDB (Fast Signal Database) format. This format, which employs a binary system,

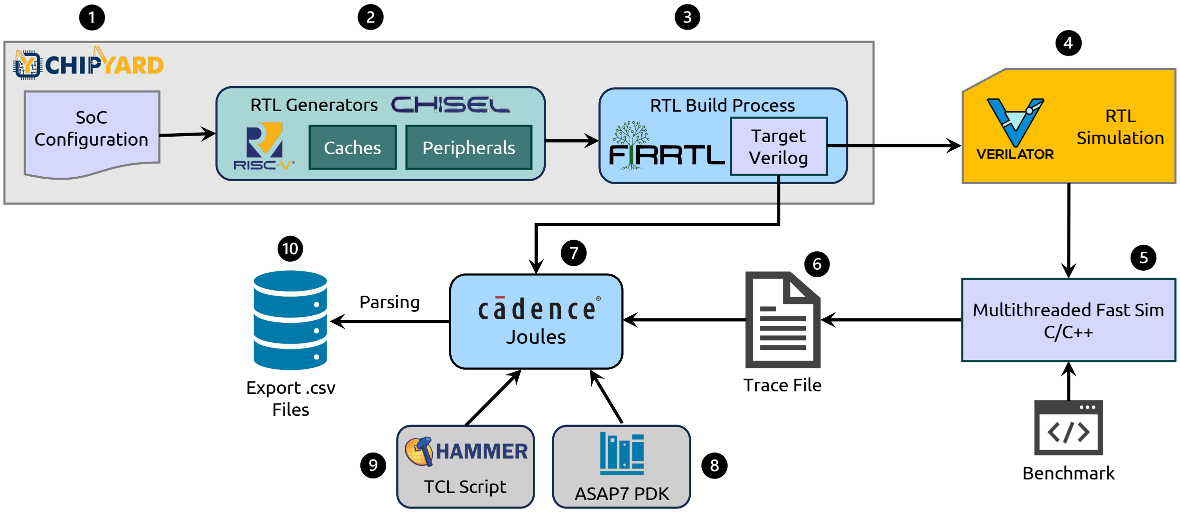

Figure 2: The experimental flow used in this study for RTL power modeling and evaluation.

TABLE 1: BOOM Configurations Used in this Study.

	Medium	Large	Mega
Fetch/Decode/Issue Width	4/2/4	8/3/5	8/4/8
Int / Mem / FP IQ Entries	20/12/16	32/16/24	40/24/32
Int / Mem / FP EXUs	2/1/1	3/1/1	4/2/2
Int Regs	80	100	128
Int RF Rd Ports	6	8	12
Int RF Wr Ports	3	4	6
FP Regs	64	96	128
FP RF Rd Ports	3	3	6
FP RF Wr Ports	2	2	4
ROB Entries	64	96	128
L1 I$ Size / Associativity	16KB / 4	32KB / 8	32KB / 8
L1 D$ Size / Associativity	16KB / 4	32KB / 8	32KB / 8
Load / Store Queue Size	16 / 16	24 / 24	32 / 32
Branch Predictor	GShare	GShare	GShare
BTB Entries	256	512	512

significantly reduces the file sizes—by as much as tenfold. Such a reduction streamlines our processes and makes the subsequent steps in our methodology more efficient.

After obtaining the required trace files, we feed them to Cadence Joules ❼ along with the target Verilog RTL code produced in a previous step along with the ASAP7 PDK library files ❽ and HAMMER generated TCL script ❾ that runs the Joules power-estimation flow as described in Section 3.2. In order to study the effects of microarchitecture on power consumption, we use the same clock rate in all three BOOM configurations (therefore, the microarchitectures differ in their IPC—instructions per cycle). To better fit our needs, we made several modifications to the default flow. We made adjustments to the generated script to eliminate the need for performing a new synthesis every time we need to execute power estimation for a different benchmark on a specific configuration. Instead, we save the synthesized database and restore it when it exists, bypassing the time-consuming re-synthesis process. Moreover, we enable multithreaded reading of the trace files, resulting in vastly quicker stimulus processing. Finally, we implemented customized power reporting commands into the flow. This facilitates our results parsing ❿ using a tailor-made Python script, enabling us to extract and interpret our findings in a more organized and comprehensive manner.

3.2. Joules Power Estimation

Initially both RTL source (Verilog [15], SystemVerilog [16], VHDL [17]) and Process Design Kit (PDK) library files (Liberty, LEF) [9] are read by the tool. The RTL source is elaborated and saved in a database to be used in the Design Mapping (Synthesis) and Stimulus (Trace File) Processing stages. During the technology mapping stage, a rough synthesized netlist is created by implementing the arbitrary RTL constructs with actual library standard cells. This process is much faster than an actual full chip synthesis step. Joules takes into account the PDK liberty files that contain timing, area, and power specifications of standard cells and constraint files that define the desired clock frequency and other important design constraints. Trace files from RTL simulators are used to determine the toggle rate of each signal in the design. Joules can accept a variety of trace file formats that depending on workload length can range from a few Megabytes to hundreds of Gigabytes of data. Finally, power-estimation is performed by first generating an appropriate clock-tree and then using the mapped netlist in conjunction with the calculated toggle rate of each signal to determine the Leakage, Internal, and Switching power of the design.

4. Experimental Results & Analysis

In this section, we focus on analyzing the results derived from the experimental methodology discussed in Section 3. We aim to thoroughly evaluate the obtained data and gain valuable insights on how microarchitecture impacts power consumption and how the power of each hardware component scales as its size varies and observe the workload sensitivity of each component's power consumption.

4.1. Power Consumption Analysis per Microarchitectural Component

Figures 3, 4 and 5 show the power consumption of each of the 13 hardware components considered in this study across all ten workloads for the three BOOM configurations, as described in Section 3.

Due to space limitations we provide insights about the BOOM scheduler. Similar observations can be made for the rest of the microarchitectural components. BOOM employs a three-way distributed scheduler design, where different units are responsible for issuing different types of instructions. The Integer Issue Unit handles integer instructions, the Memory Issue Unit handles memory instructions, and the Floating Point Issue Unit handles floating-point instructions. Among these units, the Integer Issue Unit exhibits the highest power consumption across all BOOM configurations, as shown in Figures 3, 4 and 5. On average for all benchmarks, the Integer Issue Unit consumes 0.73mW in MediumBOOM, 1.56mW in LargeBOOM, and 2.67mW in MegaBOOM. These values correspond to 6.7%, 8.4%, and 8.5% of the total power consumed by the BOOM tile, respectively. The Integer Issue Unit demonstrates significant workload sensitivity, as its power consumption is closely tied to the ILP of the executed program. A high number of independent integer instructions leads to increased activity within the issue unit, resulting in high power consumption. This correlation explains why the integer issue unit's power consumption varies depending on the workload.

4.2. Performance per Watt

Performance per watt is a metric used to measure the efficiency of a system or device by assessing its performance relative to the amount of power it consumes. It is commonly used in the context of computer systems, processors, graphics cards, and other electronic devices where energy efficiency is important. Performance per watt is typically expressed as a ratio or a figure of merit, such as instructions per watt (IPC/W). These metrics indicate how much work or computational capability can be achieved per unit of

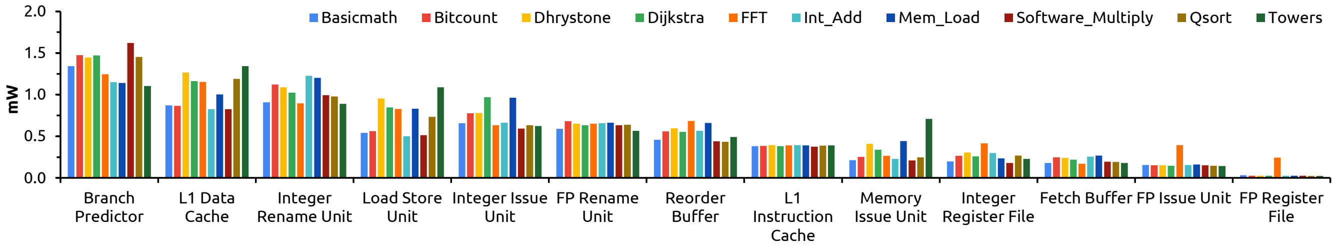

Figure 3: Power consumption per hardware structure for the MediumBoom across all ten benchmarks.

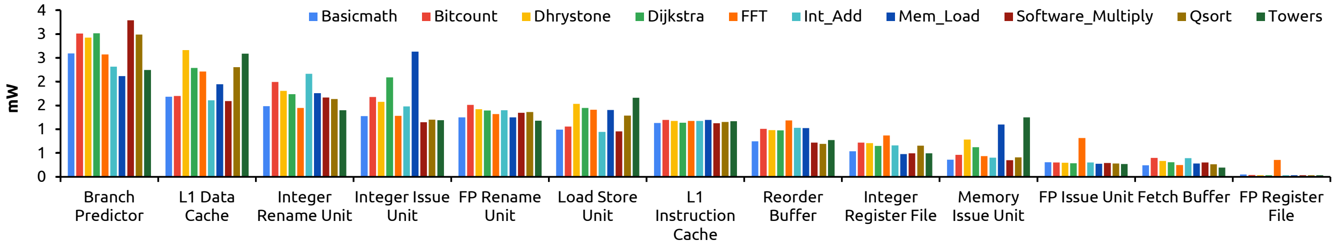

Figure 4: Power consumption per hardware structure for the LargeBoom across all ten benchmarks.

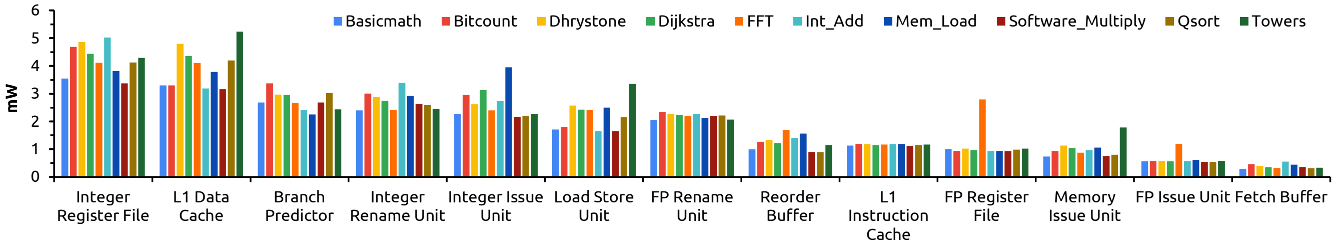

Figure 5: Power consumption per hardware structure for the MegaBoom across all ten benchmarks.

electrical power consumed. The importance of performance per watt has increased significantly in recent years due to the growing demand for energy-efficient computing systems. As power consumption has become a limiting factor in many applications, improving performance per watt has become a key goal for hardware manufacturers. Figure 6 shows the performance per-watt for all three BOOM configurations across the 10 benchmarks we executed (the right-most bars show the average values for all benchmarks of each configuration). We observe that in 8 out of 10 benchmarks, the MediumBOOM configuration prevails. In the *Bitcount* and *Int_Add* benchmarks, LargeBOOM marginally takes the lead. MegaBOOM while having the best absolute performance sacrifices significantly more power to achieve it.

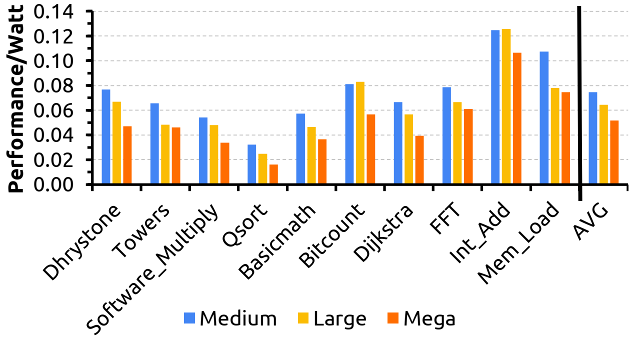

Figure 6: Performance per Watt for each benchmark (x-axis) and for all three BOOM configurations.

5. Conclusion

We presented an exploration of the impact of microarchitectural configurations on energy efficiency (i.e., power consumption and performance) using the state-of-the-art RISC-V based CPU design (SonicBOOM). We demonstrated how microarchitectural design choices significantly influence power consumption profiles. By assessing the power consumption of three different configurations of the SonicBOOM design and evaluating the effects of thirteen major microarchitectural components, we gained insights to the relationship between microarchitecture and power consumption. Furthermore, our performance per watt evaluations showed that, despite providing lower performance, smaller RISC-V designs exhibited higher performance per watt ratios (i.e., thet are more energy efficient). These findings have important implications for system designers and computer architects, as they can now make informed decisions to optimize energy efficiency in modern microarchitectures.

Acknowledgments

This work is supported by research gifts from Intel and AMD, as well as the European Union's Horizon Europe research and innovation programme under grant agreement No 101070238 (Neuropuls). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

References

[1] D. Brooks, P. Bose, S. Schuster, H. Jacobson, P. Kudva, A. Buyuktosunoglu, J. Wellman, V. Zyuban, M. Gupta, and P. Cook, "Power-aware microarchitecture: design and modeling challenges for next-generation microprocessors," *IEEE Micro*, vol. 20, no. 6, pp. 26–44, 2000.

[2] P. Koutsovasilis, C. D. Antonopoulos, N. Bellas, S. Lalis, G. Papadimitriou, A. Chatzidimitriou, and D. Gizopoulos, "The impact of cpu voltage margins on power-constrained execution," *IEEE Transactions on Sustainable Computing*, vol. 7, no. 1, pp. 221–234, 2022.

[3] Khubaib, M. A. Suleman, M. Hashemi, C. Wilkerson, and Y. N. Patt, "Morphcore: An energy-efficient microarchitecture for high performance ilp and high throughput tlp," in *2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, 2012, pp. 305–316.

[4] A. Gamatie, G. Devic, G. Sassatelli, S. Bernabovi, P. Naudin, and M. Chapman, "Towards energy-efficient heterogeneous multicore architectures for edge computing," *IEEE Access*, vol. 7, pp. 49 474–49 491, 2019.

[5] P. R. Bodmann, G. Papadimitriou, R. L. Rech Jr., D. Gizopoulos, and P. Rech, "Soft error effects on arm microprocessors: Early estimations versus chip measurements," *IEEE Transactions on Computers*, vol. 71, no. 10, pp. 2358–2369, 2022.

[6] J. Zhao, B. Korpan, A. Gonzalez, and K. Asanovic, "Sonicboom: The 3rd generation berkeley out-of-order machine," in *Fourth Workshop on Computer Architecture Research with RISC-V*, vol. 5, 2020.

[7] A. Amid, D. Biancolin, A. Gonzalez, D. Grubb, S. Karandikar, H. Liew, A. Magyar, H. Mao, A. Ou, N. Pemberton *et al.*, "Chipyard: Integrated design, simulation, and implementation framework for custom socs," *IEEE Micro*, vol. 40, no. 4, pp. 10–21, 2020.

[8] K. Asanovic, D. A. Patterson, and C. Celio, "The berkeley out-of-order machine (boom): An industry-competitive, synthesizable, parameterized risc-v processor," University of California at Berkeley Berkeley United States, Tech. Rep., 2015.

[9] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthy, and G. Yeric, "Asap7: A 7-nm finfet predictive process design kit," *Microelectronics Journal*, vol. 53, pp. 105–115, 2016.

[10] W. Snyder, "Verilator: Open simulation-growing up," *DVClub Bristol*, 2013.

[11] ——, "Verilator 4.0: open simulation goes multithreaded," in *Open Source Digital Design Conference (ORConf)*, 2018.

[12] "Cadence joules," https://www.cadence.com/en_US/home/tools/digital-design-and-signoff/power-analysis/joules-rtl-power-solution.html, accessed: 2023-06-12.

[13] H. Ranjitha, S. Hiremath, and S. G. Langadi, "Rtl power estimation: Early design cycle approach for soc power sign-off," in *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*. IEEE, 2018, pp. 480–484.

[14] M. R. Guthaus, J. S. Ringenberg, D. Ernst, T. M. Austin, T. Mudge, and R. B. Brown, "Mibench: A free, commercially representative embedded benchmark suite," in *Proceedings of the fourth annual IEEE international workshop on workload characterization. WWC-4 (Cat. No. 01EX538)*. IEEE, 2001, pp. 3–14.

[15] D. Thomas and P. Moorby, *The Verilog® hardware description language*. Springer Science & Business Media, 2008.

[16] S. Sutherland, S. Davidmann, and P. Flake, *SystemVerilog for Design Second Edition: A Guide to Using SystemVerilog for Hardware Design and Modeling*. Springer Science & Business Media, 2006.

[17] P. J. Ashenden, *The designer's guide to VHDL*. Morgan Kaufmann, 2010.