A Survey of FPGA Optimization Methods for Data Center Energy Efficiency

Mattia Tibaldi and Christian Pilato, *Senior Member, IEEE*

***Abstract*—This article provides a survey of academic literature about field programmable gate array (FPGA) and their utilization for energy efficiency acceleration in data centers. The goal is to critically present the existing FPGAs energy optimization techniques and discuss how they can be applied to such systems. To do so, the article explores current energy trends and their projection to the future with particular attention to the requirements set out by the *European Code of Conduct for Data Center Energy Efficiency*. The article then proposes a complete analysis of over ten years of research in energy optimization techniques, classifying them by purpose, method of application, and impacts on the sources of consumption. Finally, we conclude with the challenges and possible innovations we expect for this sector.**

***Index Terms*—Sustainable computing, Data centers, Cloud, FPGAs, Power optimizations, PUE**

I. Introduction

The total amount of data generated globally is rapidly increasing and expecting to reach 180 zettabytes by 2025 [1]. This trend is due to several factors. For example, in 2020, the amount of data created and replicated reached a new high due to the *COVID-19* pandemic and the boost of smart-working and contact tracing [2]. Nowadays, this massive amount of data is processed almost entirely in large data centers, generating up to 2% of the global CO_2 emissions [3], [4]. Since one of the key challenges for the next years is certainly climate change, the *information technology (IT)* sector must necessarily contribute to reducing emissions. In this context, it is no longer possible to ignore the data center's contribution, and analyzing the impact of modern technologies is extremely important to make educated decisions for their sustainable development [5].

A data center is a physical structure where hundreds to thousands of servers are allocated, organized, and managed to provide specific no-stop services. Data centers can be classified by the type of architecture, i.e., *traditional*, *cloud*, and *hyperscale*, and by the *tier* level, i.e., from 1 to 4, based on their characteristics, like uptime guarantee, downtime per year, redundancy, concurrently maintainable, and price [6]. A *traditional* data center is a small set of IT equipment and on-premises, often in conjunction with a corporate office. A *cloud* data center combines physical servers running a single operating system with virtualized ones. In this way, a single physical server can house multiple virtual servers, increasing the efficiency and scalability of the infrastructure. When this concept reaches the limit, we have *hyperscale* data centers. Hyperscale systems can manage a large network of computers (e.g., tens of thousands) to automatically respond to changes in user requests and assign workloads on demand to computing, storage, and network resources. Figure 1 shows the consumption trends for each data center category. Hyperscale data centers are becoming predominant, and their overall energy demand reached 86.58 TWh in 2021 [7]. Instead, more and more companies are abandoning traditional data centers in favor of cloud or hyper-scale versions. Table I summarizes the major characteristics of each *tier*. We need to keep this hierarchy in mind when we compare it from a data center energy standpoint. Typically, the higher *tiers* are also the most consuming devices.

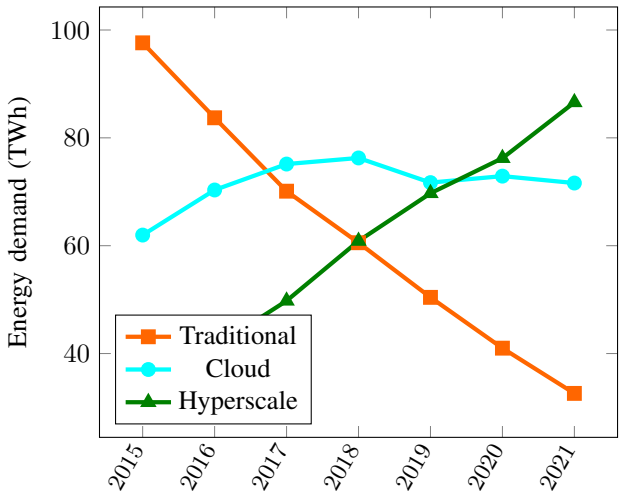


Fig. 1: Data Center power demands from 2015 to 2021 [7]

To reduce the energy consumption of data centers, FPGA is emerging as a computing technology in this field. With the Catapult project [8] in 2014, Microsoft introduced FPGAs in commercial systems. Many IT protagonists such as Alibaba, Amazon, and Huawei now support FPGAs in their data centers and make them available to application developers. Today a market of over 6 billion dollars is estimated for this technology [9]. However, their long-term adoption in these venues is not guaranteed. Their simple integration into data centers does not mean significant reductions in energy consumption. Managing them requires specific middleware, hardware virtualization, and domain separation mechanisms that make designing efficient architectures for such systems complex [10]. Also, it is not always true that FPGA consumes less than central processing unit (CPU) or other accelerators like graphic processing unit (GPU). Only a combination of specific problems with specific complexity results in immediate energy saving [11]. Several optimizations must be provided

M. Tibaldi and C. Pilato are with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milan, Italy (Contact email: *mattia.tibaldi@polimi.it*).

TABLE I: Data Center Tiers Compared

Parameters	Tier 1	Tier 2	Tier 3	Tier 4
Uptime guarantee	99.671%	99.741%	99.982%	99.995%
Downtime per year	$<$ 28,8 h	$<$ 22 h	$<$ 1,6 h	$<$ 26,3 min
Redundancy	None	Partially	Partially+	Fully
Concurrently maintainable	No	No	Partially	Fully
Price	$	$$	$$$	$$$$
Typical customer	Small companies	Medium business	Large businesses	Government entities

for FPGAs to make a sustainable system. The configuration also introduces significant challenges. Traditional energy optimization techniques require function calibrations and must re-evaluate their effectiveness. Many obstacles still exist in FPGA data center deployment. The main one is the lack of support for FPGAs in the asset management and data center monitoring tools. Besides, there is not enough software stack to allow easy deployment on the cloud. This survey analyzes this problem, discussing which solutions can be applied at each level and highlighting the open points and the expected benefits. After describing the energy targets for the next year and the *European Code of Conduct for Data Center Energy Efficiency* (Section II), and providing an analysis on the current status of the technology (Section III), we present our main contributions:

- An overview of the main metrics and methods used in data centers to analyze energy consumption (Section IV).
- A critical review of the energy optimization techniques on FPGAs and their possible integration in data centers (Section V). This part is the core of this work, summarizing over ten years of research and presenting suggestions on to apply the methods and increase their effectiveness.
- A discussion of the most used solutions and the possible innovations (Section VI and Section VII, respectively).

The entire article is structured to address and cover all the points presented in the *Code of Conduct* concerning the introduction of new technologies in data centers. We indeed use this manifest to guide the reader through the work. Finally, we conclude the article in Section VIII with a summary of the opportunities and challenges in this domain.

II. Energy Targets for the Next Years

Works like [12], [13], [14] identify and classify the biggest causes of energy consumption in data centers. The energy consumption devices can be generally classified into four categories:

- **Terminal devices**: e.g., CPUs, GPUs, FPGAs, and servers;
- **Network devices**: e.g., routers and switches;
- **Storage devices**: e.g., memories, solid-state disks (SSDs), and hard-disk drives (HDDs);
- **Environmental devices**: e.g., cooling, lighting, and power supply devices;

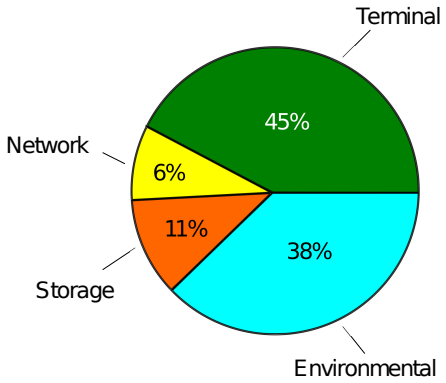


Fig. 2: Main sources of consumption in a data center.

Figure 2 shows how these categories impact data center consumption. We attribute most of the consumption to the terminal and environmental devices, representing 45% and 38% of the total absorbed energy, respectively [15]. The ideal condition is that all the absorbed energy is used entirely by the IT devices. Currently, few data centers manage to get close to this ideal condition, and they succeed only thanks to their particular geographical position. For example, the *Norwegian Green Mountain* data center [16] exploits the geological conformation of the environment and the harsh climate of the fjords as a cooling system, almost eliminating the energy consumption due to environmental devices. Unfortunately, not all data centers can benefit from such advantageous locations and other methods are needed to limit their consumption.

In 2008, a voluntary initiative, named **European Code of Conduct for Data Center Energy Efficiency program** [17], was created within the European Union in response to the increasing energy consumption in data centers and the need to reduce the related environmental, economic, and energy supply security impacts. Over 120 organizations participate in the program, and 289 data centers periodically submit complete energy data to the initiative. The *European Code of Conduct for Data Center Energy Efficiency* poses a series of guidelines and best practices to help ensure that participants are committed to a substantial energy-saving effort. They associate each solution to an application area (e.g., entire data center, new software, new IT equipment, new build or retrofit, and optional) and a value from 1 to 5 to indicate the level of benefit to be expected and the relative priorities. In the following, we report some of the guidelines for IT equipment reported with the highest priority:

❶ *Include the energy-efficiency performance of the IT device as a high-priority decision factor in the tender process;*
❷ *Include the operating temperature and humidity ranges at the air intake of new equipment as high-priority decision factors in the tender process;*
❸ *Formally change the deployment process to include the enabling of power management features on IT hardware as it is deployed;*
❹ *Select equipment that provides mechanisms to allow the external control of its energy use;*
❺ *Processes should be put in place to require senior business*

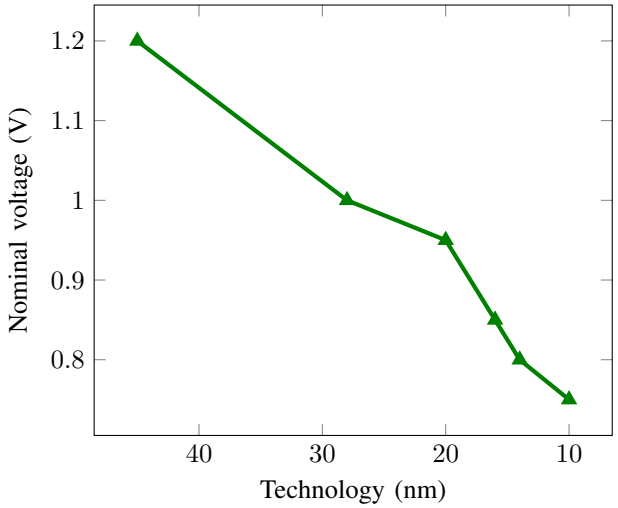


Fig. 3: Voltage trends for the main production processes.

approval for any new service that requires dedicated hardware and will not run on a resource-sharing platform;

Note that new software is reported as *fundamental* to make the energy use of the software a primary selection factor.

FPGA technology fully meets the requirements for new IT devices by the code of conduct, becoming a potential investment for most of the data centers in the area. However, their simple integration does not guarantee optimal results in reducing consumption. Specific knowledge is needed on the main existing optimization techniques to fully exploit them, and professionals from different disciplines must work together to ensure satisfactory results. On this aspect, the *Code of Conduct* focuses on the importance of establishing an "approval board" for each decision including representatives from all disciplines like senior management, IT, engineering, applications/software, and procurement. Collaboration is essential to properly understand the problem and reach an effective solution in such complex systems. This survey aims to be a reference tool for deciding whether to invest in FPGAs within data centers and is structured to respond to each caveat presented in the *Code of Conduct*. Section III highlights the current state of the FPGA technology with the latest innovations. Section IV discusses potential comparison metrics between different solutions giving hints for defining new ones. Each of the five points above (i.e., ❶ to ❺) is reflected in Section V, where we evaluate existing energy optimization techniques. For completeness, we have preferred to discuss all the existing energy optimizations on FPGAs and give an opinion on their adoption in the data center rather than limiting ourselves only to the most promising techniques. Each optimization follows two classifications: one based on the expected benefits in terms of energy savings (Table II) and one based on which area of the data center it affects (Figure 5).

III. FPGA INNOVATIONS AND ACTIVE RESEARCHES

FPGA devices are reconfigurable chips based on CMOS and static random access memory (SRAM) technologies. Their structure achieves excellent performance and low consumption. On these devices, one can create a customized implementation of an algorithm that operates on multiple data in parallel. So, it can be faster and consume less power than processors with higher clock speeds. Reconfiguration allows users to change the functionality every time is needed. So, an FPGA trade offs performance and flexibility, resulting a suitable product for data centers where the same node can serve multiple users. In this section, we highlight the main shortcomings of the technology with today's solutions and possible research areas to improve their use inside data centers.

A. The need for efficiency

The integration level of FPGA technologies is currently far from the CPU world, resulting in reduced efficiency. Great progress has been made since the first FPGA from Xilinx appeared on the market in 1985. This initial technology was with a production process at 2000 nm and the first devices had 64 configurable logic block (CLB). Today, after almost 40 years of research, we find FPGAs on the market with over 100,000 CLBs made with production processes ranging from 45 nm for AMD-Xilinx Spartan 6 [18] to 10 nm for Intel-Altera Agilex [19]. Having a clear understanding of the type of technology and its evolution is essential to identify the optimizations and challenges of the future, especially in terms of energy. Also, the *Code of Conduct* calls for considering the entire hardware manufacturing process to understand and estimate which impacts the technology will have inside the infrastructure. Different production processes correspond to different supply voltages and different consumption. Figure 3 shows the linear trend of the voltages for the main production processes, from 1.2 V for 45 nm down to 0.75 V for 10 nm. Improving the production process allows designers to reduce the consumption of these devices.

Recently, the media announced the Intel/VMware Crossroads 3D-FPGA Academic Research Center as a multi-university effort to improve the future of FPGA technology [20]. By stacking multiple FPGA dies vertically, researchers should be able to achieve a higher transistor density while also balancing performance, power, and manufacturing costs. Therefore, the research in this area is far from standing, and there are ample opportunities for improvement.

B. The need for bandwidth

Due to the proliferation of big data applications [10], the need to move greater quantities of data has resulted in increasing bus lines inside the FPGA. One of the main problems that FPGAs suffered was the latency for memory access and the poor throughput they obtained. With the recent introduction of **high-bandwidth memorys (HBMs)**, the problem has been partially addressed [21]. HBM is a 3D-stacked dynamic random access memory (DRAM) that offers high-bandwidth and energy-efficient data movements introduced in the latest generation FPGAs such as AMD Xilinx's Alveo U280 and Intel Altera's Stratix 10. This type of memory allows reaching throughput of over 460 GB/s, in the case of the Alveo U280,

allowing fast memory access and reducing the effect of super-long-line (SLL), the high latency technology currently used to access FPGA resources from any of its regions [22]. However, the HBM technology with large bus lines is difficult to be fully exploited, demanding hard-crafted solutions or advanced design methods [23], [24].

C. The need for improved clock structures

Conventional FPGA core architectures have been based on balanced clock trees, which minimize deterministic skew. This method has served well for designs up to 500 MHz, but they need innovative solutions to reach speeds of up to 1 GHz. The solution must minimize local variation and skew, and provide a flexible network to serve the numerous clock regions. For example, to address these challenges, Intel introduces an entirely new core architecture, called *Intel HyperFlex FPGA Architecture* [25]. They add additional registers in every interconnect routing segment and at the inputs of all functional blocks. With these elements, they can retime registers to eliminate critical paths, add pipeline registers to remove routing delays, and optimize the design for best-in-class performance.

D. The need for simplified development

Creating an "implementation" for FPGA consists in making a circuit that relates to the resources present in the device. The process uses hardware description language (HDL) such as Verilog or VHDL or, more commonly today, C/C++ language compiled with an high-level synthesis (HLS) compiler for the circuit definition. We can identify four steps for generating the FPGA configuration file (i.e., *bitstream*).

1) **Hardware design**: HDL codes are manually written manually or derived from high-level specifications by means of HLS compilers;
2) **Synthesis**: HDL codes are compiled and translated into netlists;
3) **Implementation and routing**: the synthesized design is mapped onto the target FPGA resources and the connections are defined among them;
4) **Bitstream generation**: the implemented design is translated into a configuration file (also called "bitstream") which can be downloaded onto the FPGA.

Using HDL languages is not simple, and realizing well-optimized projects requires substantial specific knowledge and time. Major FPGA manufacturers provide development kits that simplify this tedious and error-prone process. Both *AMD-Xilinx Vitis suite* and *Intel-Altera Quartus suite* contain HLS compilers that allows the designer to transform a program written with a high-level language like C or C++ into HDL. These tools allow designers to apply several optimizations such as pipelining and loop unrolling, making application development easier and faster [26] by means of specific *pragmas* (e.g., **#pragma hls dataflow**, **#pragma hls array partition**, or **#pragma hls inline**) that are used by the designers to annotate the code in order to specify where to apply such optimizations. In addition to these solutions, there are many open-source HLS projects such as Bambu [27], Merlin [28], and AutoDSE [29] that try to make this process even more intuitive and automated for a programmer without specific knowledge about FPGAs. Recently, there has been a growing interest in open-source compilers, not only in academia. For example, Xilinx recently released the source code of the *Vitis HLS Front-End* [30] that can help boost the innovation around HLS tools. Conversely, the technical specifications of the physical devices (including the bitstream format) are still mostly closed-source. We argue that more flexibility from commercial solutions (e.g., clear API for interaction and analysis of implemented designs) can further boost research and innovation in this area.

IV. Evolution of power metrics

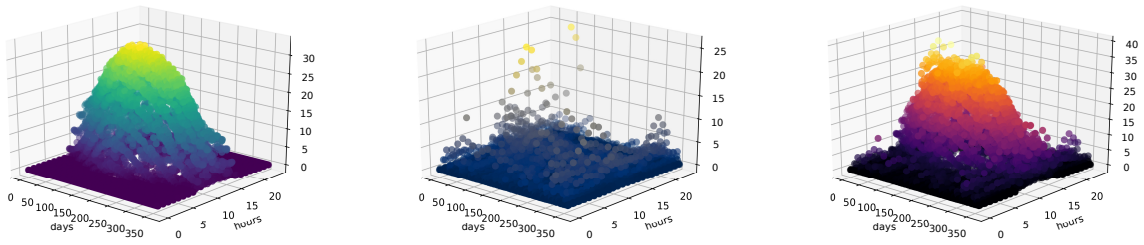
In this section, we discuss the metrics for monitoring energy consumption on the individual FPGA device and the overall data center environment. Before starting, we want to present the distinction between power and energy consumption. Measures of power are most interesting for sizing the data center, cooling systems, or power units. On the other hand, when we talk about energy, we consider the integral of power over time. The second measurement is functional from a green point of view because it can be quickly transformed into kg of CO_2 produced. The *Code of Conduct* suggests using both units to have complete visibility into the data center.

A. FPGA power metrics

Complementary metal-oxide semiconductor (CMOS) transistors are the basic blocks of FPGAs; therefore, we can divide the power consumption into two categories: *dynamic power* and *leakage power*. **Dynamic power** occurs from the switching activities because of the short-circuiting current and charging and discharging of load capacitance. It is therefore related to when the circuit is running. As anticipated in the previous section, each CLB is logic that consumes power since the power manager unit (PMU) continuously powered it. **Leakage power** is always present, and the power dissipation occurs in the form of leakage current when the system is not powered or is in standby mode. The total consumption of the device is given by the sum of these two components, as shown in Equation 1.

$$P_{fpga} = \alpha C F V^2 + g V^3 \tag{1}$$

where CFV^2 represents dynamic power, and gV^3 is the leakage one. In addition, α is the activity factor, C the capacitance, V the supply voltage, F the frequency, and g the leakage factor, which is an intrinsic factor of the device under analysis. The dynamic power has a quadratic trend with increasing voltage, while the leakage power has a cubic trend. Today, modern packaging techniques allow for high densities of CMOS per unit area, making leakage power the main factor in the equation. Further considerations are covered in Section V-A. Observing Equation 1, we can adjust only two parameters to improve energy consumption. The first is the frequency which affects only the dynamic power, and the second is the voltage which instead reduces both components. In Section V, we discuss the techniques that operate on these two parameters, how FPGAs can integrate them, and what benefits they entail.



(a) Solar energy production (b) Wind energy production (c) Combined energy production

Fig. 4: Energy production of an hypothetical medium energy park in one year. The values are expressed in MWh.

Often the results of the energy consumption of the accelerators are related to the number of floating-point operations performed in the unit of time (FLOPS), leading to the definition of the FLOPS/Watt [31] metric. This metric allows us to effectively compare different accelerators and choose the one with a higher value (i.e., the most energy-efficient one).

The model discussed above does not take into account the thermal state of the device. A chip subjected to diverse temperatures consumes a different amount of energy, and therefore, running the same software will absorb more (less) if the chip is hotter (cooled). **Thermodynamics computing** [32] tries to unify the mathematical model of the information with the thermodynamic model. The adoption and development of this model would lead to a better understanding of the phenomenon illustrated and the possibility of having energy models that are more accurate and better optimize than the current ones. The model would also be relevant from the **design of the cooling system** viewpoint [33]. We can model the capacity of the cooling system (C) as in Equation 2.

$$C = \sqrt[3]{\frac{P_{fan}}{T_{hot} - T_{cold}}} \tag{2}$$

Where P_{fan} is the maximum power absorbed by the board fans, T_{hot} is the temperature of the exhausting air from the chip, and T_{cold} is the operational target temperature, commonly set to 25 degrees. Studies also show that for every watt of power utilized during the chip operation, the cooling equipment consumes an additional 0.5-1 watts to extract the exhausting air from the IT racks [15]. This example shows how desirable a model change is and how it fits into the design of the entire data center. In conclusion, optimizations that can optimize absorbed energy and dissipated heat simultaneously are the most suitable for future developments.

B. Data center power metrics

As mentioned in section II, the causes of energy consumption in a data center are many, and we can individuate four categories: terminal devices, network devices, storage devices, and environmental devices. The sum of all these components gives the overall consumption of the data center. [17] reports the average energy consumption of 289 European data centers participating in the code of conduct, estimating an average annual electricity consumption of 13,684 MWh, of which 7,871 MWh refer only to the IT sector (e.g., terminal devices and network devices). In a data center, monitoring systems, included within the power distribution units (PDUs), continuously control consumption. Each rack has one or more dedicated PDUs with several outputs equal to the number of servers allocated (e.g., up to 42 servers per rack). In real time, it is possible to know how much energy is absorbed by a specific server, rack, or cluster (e.g., a set of racks). This solution identifies potential faults in the system and allows the technician to keep consumption under control. Consumption data are collected and processed by specific tools that monitor the progress of the infrastructure. Google is working on new tools that link consumption with environmental impact and give complete transparency to the public on what happens in a data center to raise public awareness of the issue [34].

In this work, we focus on the consumption due to the IT department and, in particular, the terminal devices. The consumption of a server can be modeled as the consumption of the individual parts [35], combined as shown in Equation 3.

$$\begin{aligned} P_{server} = & P_{cpu}u_{cpu} + P_{memory}u_{memory} + P_{disk}u_{disk} + \\ & P_{nic}u_{nic} + P_{fpga}u_{fpga} \end{aligned} \tag{3}$$

Where $P_{cpu}, P_{memory}, P_{disk}, P_{nic}$, and P_{fpga} are the power consumption of the respective components, while $u_{cpu}, u_{memory}, u_{disk}, u_{nic}$, and u_{fpga} are the utilization rate of the different components. In the ideal case, the utilization of the accelerators should be one, so it is always exploited to the maximum, reducing the utilization of more power-hungry components, such as the CPU.

A different model, instead, treats the data center similar to the one for FPGA [35], dividing the power into dynamic P_{var} and static P_{fix}. We can model it as follows:

$$P_{total} = P_{fix} + P_{var} \tag{4}$$

The idle states of the servers characterize P_{fix}, while the moments of use and the actions that occur, like the phases of data computation or movement between physical memories, characterize P_{var}.

The *European Code of Conduct for Data Center Energy Efficiency* defines, in addition to the total power consumption

in Watts, a second metric called **power usage effectiveness (PUE)** [17], introduced by The Green Grid in 2007. The PUE helps understand the data center's efficiency and how to reduce energy consumption. It is the ratio between the total data center input power and the power used by the IT equipment (e.g., Equation 5).

$$PUE = \frac{TotalFacilityPower}{ITEquipmentPower} \tag{5}$$

The ideal value is 1, which happens when the entire facility power is consumed by the IT department and not by lighting, cooling systems, power distribution units (PDUs), fans, and other equipment. Higher values imply lower efficiency. The average PUE of today's European data centers is 1.8, but the geographic location of the individual facility is important. For example, thanks to its particular structure and position, the *Green Mountain* data center [16] has a PUE of 1.2.

Concerning point ❶, the *Code of Conduct* suggests the introduction of new metrics that distinguish between the energy consumed from renewable and fossil sources. Renewable energy is not continuous in time and can assume decidedly different values according to the period. For example, consider a solar panel. It will produce more during the summer and sunny hours, rather than during the winter and nights. Figure 4 shows this variability in the two principal renewable sources exploited today by studying the energy production of a hypothetical wind and solar farm on an annual and daily scale. Figure 4c shows the total renewable energy entering the data center as a combination of the two production sites. An interesting metric could consider these production curves and estimate the over-consumption (i.e., how much non-renewable energy the data center uses) and the under-consumption (i.e., how much renewable energy the data center is produced but not used). Equation 6 defines this metric:

$$M(E_u) = \begin{cases} \alpha \frac{E_r - E_u}{E_r}, & \text{if } E_u \leq E_r \\ \beta \frac{E_u - E_r}{E_u}, & \text{if } E_u > E_r \end{cases} \tag{6}$$

Where E_u is the instant used energy, E_r is the instant renewable energy, and $\alpha|\beta$ are two parameters between 0 and 1 to scale the relevance of the contribution. The data center better uses the available energy when M is close to 1. This metric can be combined with adaptive systems that regulate accelerator energy consumption (see Section V-C).

The metrics that we have presented do not consider the consumption due to the construction of the data center and its components, as well as do not relate energy consumption with environmental metrics such as the global warming potential (GWP) [36] [37] [38]. A model that considers all these aspects is the **life cycle assessment (LCA)**. LCA is a powerful tool applicable to the data center sector [39]. In latest years, some works have been carried out on the subject. [40] present a complete LCA from cradle to grave of a UK data center. Their work highlights how the impact on the environment of a data center is completely attributable to production and the use of IT equipment (mainly servers). They also observe how the gap between the impacts of production and use decreases in the states where energy production has greener.

TABLE II: Expected benefits on the entire data center by applying each energy optimization. ✻✻✻ identifies techniques that guarantee substantial energy savings and are directly connected to the *Code of Conduct*; ✻✻ identifies complementary solutions for data centers with moderate energy savings, while ✻ identifies optional techniques.

Physical level	
Dynamic voltage scaling	✻
Adaptive voltage scaling	✻
Dynamic frequency scaling	✻
Power gating	✻✻
Remodeling	✻✻✻
Register transfer level	
Clock gating	✻✻
Efficient routing & placement	✻
Leveraging thermal margin	✻✻
Application logic level	
Approximate computing	✻✻
Deployment level	
Off-the-shelf vs Custom	✻✻✻
Centralized vs Distributed	✻✻✻
Infrastructure level	
FPGA resource virtualization	✻✻✻
Accelerated virtual machine	✻✻
Accelerated job scheduling	✻✻
Reconfiguration	✻✻

[12] proposed a similar study on a Swedish data center, concluding that new emerging technology can improve the environmental performance of the IT equipment. Currently, in the literature, no work proposes an LCA on FPGAs, reducing only to emphasizing their energy benefits.

V. Power Optimization Techniques

While there are several techniques to regulate the energy consumption of FPGAs, not all of them are suitable or attractive in data centers. In this section, we present the fundamental solutions classified by levels, from the lowest *physical level* to the highest *infrastructure level*, analyzing their relevance for the data center. For each subsection, we present a summary table of the techniques described with the substantial works that implement them and a complete discussion on how the optimization impacts the overall data center infrastructure and power consumption. Figure 5 summarizes the energy optimization techniques described in the section, highlighting the categories of energy consumption in which they affect, while Table II classifies the techniques by importance. In particular, we assign ✻✻✻ to techniques that guarantee substantial energy savings and are directly connected to the *Code of Conduct*, ✻✻ to complementary solutions implementable in data centers with moderate energy savings, and ✻ to optional techniques.

A. Physical level

In Section IV, we discussed the main metrics to measure the energy consumption of FPGAs, defining the dynamic and the leakage power. For chips made with production processes below 90 nm, it turns out that the leakage component prevails

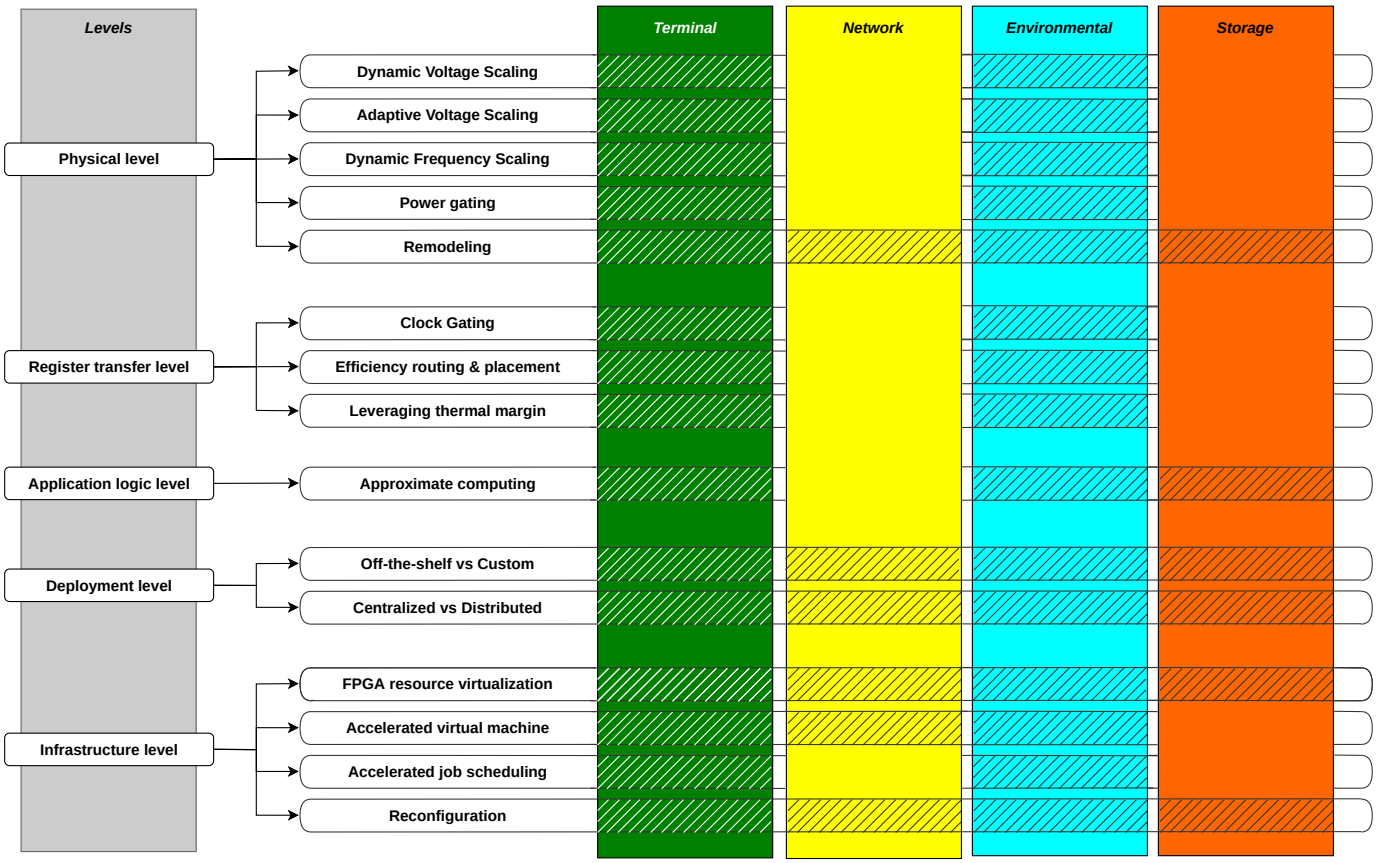


Fig. 5: Summary of the energy optimization techniques described in section V. For each technique, the categories of consumption on which they affect, both negative or positive influence, have been highlighted through the lined background of the boxes.

over the dynamic one. The smaller the process is, the more the leakage component is overcome [41]. For today's products, optimizations that work on leakage power will be more effective than those that operate on dynamic power. Unfortunately, the techniques that can adapt to reduce the leakage power are few, and the most advantages are applicable only at a low level since the leakage power is inseparable from the technology used. At the physical level, we find the following energy optimization techniques, all of them are directly linked to point ❶:

- *Dynamic voltage scaling (DVS)* allows to manipulate the voltage with which the FPGA is powered. Requires careful calibration at design time;
- *Adaptive voltage scaling (AVS)* is based on the same idea as DVS but the calibration takes place at run time with a monitoring system;
- *Dynamic frequency scaling (DFS)* allows to adjust the frequency and set it to its optimal value;
- *Power gating* turns off logic when not in use;
- *Remodeling* redesigns how the FPGAs are internally structured with an optical push for energy optimization;

Dynamic voltage scaling involves reducing the supply voltage of a circuit. It can reduce both dynamic and leakage current, but at the expense of increasing circuit delay, which can lead to timing violations and, in turn, errors [42] [43]. This effect happens because the voltage acts as the "accelerator pedal" for the signals to propagate. If we have a low voltage and a high clock frequency, parts of the circuit may work with the wrong values because of the delay. For best results, the final design should operate at the voltage that reduces power consumption as much as possible while maintaining the circuit working. Finding this threshold is problematic since the optimum operating voltage changes with time and between devices. For this reason, the system must be carefully calibrated to not run into problems. DVS is a best practice in application-specific integrated circuit (ASIC) design where the circuit does not change, and there are no further complications (e.g., timing violation due to the modification of the electric schema and a wrong DVS configuration). In FPGAs, on the other hand, more attention is needed since re-configuring the device also requires a DVS re-calibration to avoid timing problems with the risk of having an unreliable circuit. Chow et al. and Qi et al. [44] [45] propose a methodology for supporting DVS on commercial FPGAs. Their idea estimates the delay of critical paths by implementing a logic delay measurement circuit (LDMC) inside the FPGA. The module, through feedback signals, interfaces directly with the PMU to regulate the FPGA voltages. This solution effectively reduces energy consumption by up to 54% with negligible resource consumption. One advantage of their methodology is that it does not require additional design effort or changes to the FPGA itself. The main limitation of this approach is that it

requires experimentation to find appropriate threshold values for each FPGA, and it can markedly lengthen development time. Ahmed et al. [46] reduce the time due to the calibration of the LDMC by developing an automated tool (FRoC). FRoC ensures that the calibration process is invisible to FPGA users and does not add any extra manual steps to the design process. The tool generates a calibration table to scale the voltage while the application is active. A similar approach is also demonstrated by the authors of Nunez-Yanez et al. [43]. Unlike the others, they apply the reverse procedure. First, they reduce voltage and then search for the correct operating frequency. Significant savings in power and energy are measured from the nominal value for the Virtex-4 Xilinx FPGA of 1.2 V down to its limit of 0.9 V.

A different approach to the problem is given by **Adaptive Voltage Scaling**. In this case, it eliminates the calibration problem by introducing a monitor to measure the performance variability of the silicon. Works such as those proposed by Nabina and Nunez-Yanez [47] [48] show how these systems operating in a closed-loop configuration can significantly improve energy profiles compared with DVS. Implementing this control system directly within the FPGA can limit the available resources. To get around the problem, Nunez-Yanez implements the control and monitoring system on an external PCB connected to the FPGA board. The results are good, and it will be useful if FPGA vendors move towards making their devices more friendly to AVS techniques by adopting different power planes and introducing more robust built-in delay monitoring circuits. The works we have just presented do not consider applications that use the FPGA hard blocks such as block RAMs (BRAMs). Zhao et al. [49] extend a previously proposed offline calibration-based DVS approach to enable DVS for FPGAs with BRAMs. Again the idea is to run a series of tests to ensure that all used BRAM cells operate safely while scaling the supply voltage.

An optimization that always goes hand in hand with DVS is **dynamic frequency scaling**, leading to the combined acronym dynamic voltage frequency scaling (DVFS). As we have seen, when we applied DVS, it can run into timing problems due to the high frequencies that the circuit should work. Adding DFS serves to solve this problem, thus, being able to optimize voltage and frequency together. DFS can adjust the frequency and set it to its optimal value. The addition of DFS to DVS results in additional linear power savings [50]. Zhao et al. [51] presents an offline self-calibration scheme, which automatically finds the FPGA frequency and core voltage operating limit. Their idea is to explore two phases, one to estimate the actual minimum frequencies at which the logic can operate and one to generate the configuration tables for the LDMC module. These phases require one reprogramming of the FPGA each, and a power saving of around 40% justified the additional time spent in the design phase.

No doubt, the most effective technique for reducing leakage power on a chip is **power gating**. The idea is simple, turn off logic when not in use. Often, the literature uses the term *dark silicon* to refer to areas of the chip switched off [52]. The power gating technique inserts a power switch unit, controlled by an on/off signal, on the logic to optimize. The power switch's inclusion introduces overhead on the resources and the energy. The fundamental challenge for any power gating technique is to ensure that the saved leakage power outweighs the power overhead of the power gating. We can identify two different implementation styles, the *fine-grain* (divides the circuit into small sections) and the *coarse-grain* (divides the circuit into large sections) technique. The two styles are usually combined to seek a trade-off between resources used and energy saved [53]. The more finely implemented the control system is, the more the overhead will be relevant because it is necessary to insert more power switches in the circuit. Finding the right compromise can be difficult. In FPGAs, we can safely turn off all the unused resources by a specific design without any risk. It is always possible to perform this optimization with negligible overhead since for how the placement and routing phases on FPGA are made, the logic consumed is allocated in a single contiguous region, outlining a clear separation between used and unused sections. Therefore, by inserting a single power switch, you can turn off all the unused areas of the FPGA by the implemented design [54].

There are particular designs where the *coarse-grain* style has a greater area overhead than *fine-grain*. This overhead happens when the logic zones we want to control are very far from each other, resulting in long paths that add overhead in the signal distribution network [55]. On the other hand, in *fine-grain* power gating, the higher level of control allows to reduce the leakage power more. Still, since the power switches are always running, they increase the consumption of the dynamic one. Due to these overheads, *fine-grain* power gating is less efficient than coarse-grain power gating [54] [56] [57].

Another challenge with power gating is to reactivate the switched-off logic without creating delay when needed. This consideration is fundamental when we work with stream applications where the stream must proceed without interruption. A possible solution is to add a series of checkpoints along the computation phases of the streams to reactivate the logic in time [58]. DVS, AVS, and power gating are techniques not always applicable to commercial FPGAs. To implement these techniques, the PMU must be accessible and configurable from an external component such as a CPU or an ASIC associated with the FPGA. The level of control we have over the FPGA and how we can implement these techniques depends on the PMU structure. Many of the papers presented propose to **remodel** FPGAs to implement energetic optimization techniques more effectively. Qi et al. [59] redesign the FPGAs internal structure with an optical push for energy optimization. They offer a custom FPGA that includes native solutions of DVFS and Power gating, eliminating many of the previously discussed issues (e.g., interconnect delay, calibration, and so on). Today, we are shifting towards custom FPGA versions, also in relation to point ❸ and ❹. Few other techniques consider these aspects and these new architectures will be taken into consideration more and more. This increase can be seen both in the embedded systems field and in the high performance computing (HPC) field. Projects such as *OpenFPGA* [60] facilitate these new architectures prototyping. *OpenFPGA* allows users to customize their FPGA architectures down to

TABLE III: Physical level - Power optimization techniques

Techniques	Main works	Target power	
		Dynamic	Leakage
Dynamic voltage scaling	[46] [44] [42] [61] [43] [45]	✓	✓
Adaptive voltage scaling	[49] [47] [48]	✓	✓
Dynamic frequency scaling	[61] [50] [51]	✓	✓
Power gating	[53] [58] [55] [54] [52] [56] [57]	✗	✓
Remodeling	[59] [60]	✓	✓

circuit-level details using a high-level architecture description language and to autogenerate associated Verilog netlists which can be used in a backend flow to generate production-ready layouts. We expect this trend to increase, and in the future, we will probably have FPGA models with specific targets for the use sectors. Table III summarise the optimization techniques and the main works presented in this subsection.

OVERALL IMPACTS CONSIDERATION: *The optimization techniques presented in this subsection have the main purpose of limiting the consumption due to leakage power, acting on the voltage of the FPGA and the working frequency of the circuit. The primary effect obtained is reducing the P_{fpga} and, in turn, the energy consumed by the device terminals. The impacts of these techniques on the environment are not easy to deduce without a unified model with thermodynamics that considers FPGAs. Based on the Equation 2 and the approximation for computer room air conditioning (CRAC) cooling systems which says that for every 1 Watt saved on the chip, we also save about 1 Watt from the cooling system [33], we can still conclude that there is an impact and that it is positive. However, even though these techniques impact both ❶ and ❷, we believe that DVS, AVS, and DFS are valuable solutions for embedded systems but not for data centers. Therefore, we prefer to consider them optional. On the other hand, power gating is different, which we think is a good practice and which the new generation synthesis tools are adopting by default. As far as the remodeling technique is concerned, a broader discussion is needed since it could also impact the network and storage component based on how the re-modeling is carried out. A remodel pushed towards network management by inserting specific resources for the network inside the FPGA could have a significant impact on the network devices. In contrast, a remodel that leads the FPGAs to have more types and storage capacities could favorably impact the second component. Currently, there are no studies relating the remodel to the consumption categories. Still, we consider it an extremely good technique that touches on every aspect listed in the code of conduct.*

B. Register Transfer Level

Register transfer level (RTL) is an abstraction level for defining the digital portions of a design. It is the principal abstraction used for specifying electronic systems today. HDL like Verilog and VHDL uses RTL to create high-level representations of a circuit, from which specific tools can derive lower-level abstraction, now integrated into the most common development kits such as Vivado and Quartus. Thanks to this level of abstraction, it is possible to conduct an early-stage dynamic power analysis [62]. The optimization techniques available at this level focus on reducing the dynamic power (point ❶), and they operate without having to access specific components external to the FPGA, as happens instead for the physical layer that needs the PMU. The dynamic power is due to the switching activity of the signals, introducing a continuous charge and discharge of the parasitic capacitance present in the circuit [63]. Typically the signal with the highest switching activity is the clock signal. This sentence is especially true when we consider synchronous digital systems. At the register transfer level, we find the following energy optimization techniques:

- Clock gating (CG): disables the clocking of specific registers when the outputs of those registers are stable;
- Energy-efficient routing and placement: improve the interconnections between FPGA resources to guarantee energy savings;
- Leveraging thermal margin (LTM): tries to exploit the FPGA temperature to drop further the voltage;

There is one principal technique that acts on the switching activities, and it is **clock gating**. As the term suggests, clock gating is a way to reduce switching activity on circuit signals by disabling the clocking of specific registers when the outputs of those registers are stable [64] [65]. The idea of the clock-gating technique for ASICs has developed in the late nineties. Studies indicate that the clock signals in digital computers consume 15-45% of the system power [66]. Because of its effectiveness, clock-gating has been a hot topic in many research areas [67] [68] [69]. Modern FPGAs include several clock control blocks that allow to shutting down of the clock line in some parts of the circuit, and to emulate the functionality of the gated clocks is possible to use feedback multiplexers [70]. Pandey et al. [71] and Liong Tan et al. [72] implement an energy-efficient arithmetic logic unit (ALU) on FPGA. Both authors agree on the impossibility of adopting the traditional ASIC style to implement CG with AND gate because it will create a glitch in output. In modern FPGA devices, two buffer types can replace the AND gate and ensure glitch-less output: Global Clock Buffer (BUFGCE) and HROW Clock Buffer (BUFHCE). Building CG using these buffers results in an overhead that includes extra control logic to generate CG control signal and extra leakage power consumption. However, the technique saves 40% of the dynamic power [73]. The same idea seen for clock gating can also be applied to other design-dependent signals and even to the memory structure, as shown by the works of Sterpone et al. [74] and Agrawal et al. [75]. Geier et al. [76] present a module that can easily be added to current designs with memory-mapped AXI3 or AXI4Stream (AXI4S) interfaces to monitor interface signals and limit switching activity. The module insertion is completely transparent to the design and provides a quick method of applying clock gating on FPGAs.

Specific tools such as the one presented by Zhang et al. [77] or the one developed by Siemens [78] can even perform automatically clock gating optimization. For example, *powerPro* [78], starting from an RTL file and a set of vector tests runs a series of simulations to verify in the design when and where it is possible to disable the clock signal. The execution times depend on the design size and can range from a few minutes to several hours, but they are acceptable given the results. Research into automating these optimizations is critical today. With increasingly stringent power consumption requirements, they help to provide solutions already in the early stages of implementation.

Another technique that always acts on the circuit is **energy-efficient routing and placement**. This technique improves the interconnections between FPGA resources to guarantee energy savings [79] [80]. Essentially, the closer the resources are to each other, the less energy is consumed. Exploiting some design rules like avoiding the use of SLL or fast tracks and using local interconnection can save power dissipation in FPGAs [81]. Zemani and Esmaili [82] try to do just that by extending the versatile place and route (VPR) routing algorithm [83] with one more iteration where power is optimized together with area and circuit performance. Adding different performance metrics such as those reported in Section IV can extend the exploration algorithm. Something similar is also done by the Hao Hoo team [84] and Leming and Nepal [85], with the only difference being that they propose a new version for the embedded switch box (SB) of the FPGA. Seifoori et al. [86] are the first to use a machine learning technique to design power gating regions in the FPGA routing network. They define similarity metric, cluster pattern, and power gating efficiency to design three clustering algorithms based on K-means clustering. Finally, they evaluate the obtained design on an Intel Stratix-IV FPGA, obtaining 1.4× higher savings to other heuristics. Generally, an exploration algorithm based on the power consumption model combined with the place and route tool can help to find the optimal solution [83]. Chtourou et al. [87] instead of focusing on routing algorithms analyze two different routing architectures: the *SB_tristate* and the *SB_multiplexer*. The first uses bidirectional SB implemented with back-to-back tri-state drivers. The second uses bidirectional SB implemented using tri-states and multiplexers. They conclude that *SB_multiplexer* has a significant impact on power saving compared to *SB_tristate*. This difference happens because *SB_tristate* always uses the switch in only one direction, thus increasing the amount of the leakage of power dissipated by routing resources.

The last technique explained at the register transfer level is **leveraging thermal margin**. Compared to the other optimizations presented, LTM is much more recent and is a cross between physical layer techniques such as DVS and energy-efficient routing and placement [88] [89]. The most recent work is Khaleghi et al. [90] in 2019. The basic idea is that a chip at different temperatures exhibits different voltage limits, and they use this gap to optimize the voltage. If the chip is "cold" (e.g., 40 degrees), it is possible to drop further in voltage compared to one that is "hot" (e.g., 100 degrees). It is necessary to organize the logic on the FPGA and thus improve the existing routing and placement algorithms to avoid excessive temperature peaks. The technique in its current state is still complicated and requires several simulations (e.g., delay simulation, FPGA architecture simulation, and thermal simulation) before arriving at a stable solution. Furthermore, if you want to use an adaptive approach at run time, it requires access to temperature sensors that are not always available. However, the technique is promising, improving DVS up to 30%. Table IV summarise the optimization techniques and the main works presented in this subsection.

TABLE IV: RTL - Power optimization techniques summary

Techniques	Main works	Target power	
		Dynamic	Leakage
Clock gating	[76] [75] [67] [64] [65] [68] [70] [73] [71] [74] [72] [66] [69]	✓	✗
Efficient routing & placement	[87] [85] [86] [80] [83] [82]	✓	✓
Leveraging thermal margin	[88] [90] [89] [84]	✓	✓

OVERALL IMPACTS CONSIDERATION: *Considerations similar to those made previously on the impact on terminal devices and environmental devices are also valid for the techniques set out in this subsection. The work done by Khaleghi et al. [90] allows us to expand the discussion on the FPGA's impact on environmental devices. Looking at the thermal resistance θ_{JA}, we note that for today's Intel and Xilinx FPGA products, a value θ_{JA} of $2°C/Watt$ and a pessimistic thermal resistance of $12°C/Watt$ are considered. If now we combine these considerations with equation 2, we can model T_{hot} as $P_{fpga} \times \theta_{JA}$ and obtain a consistent model to estimate the cooling capacity required starting from a simple energy consumption profile. This consideration acquires importance, given points ❷ and ❹, by providing innovative metrics and relevant considerations for the dimensioning of cooling systems and data centers in general. For this reason, we consider LTM a technique that can influence more the data center than efficient routing & placement but close to clock gating since tools like [77] already adopt it as a standard. Maybe in the future, thermal optimizations will be more extensively implemented into synthesis tools.*

C. Application Logic Level

By "Application Logic Level" optimizations, we refer to all the techniques that modify the application algorithm or the input/output interfaces to achieve energy efficiency (point ❶). In general, code optimizations at a high level have a greater impact on performance than low-level optimizations [91]. A modification to the algorithm can generate a very different circuit and exhibit a different energy profile. These techniques go by the name of **approximate computing (AC)**. Approximate computing is based on the intuitive observation that while performing an exact computation requires a high amount of resources, granting selective approximation can provide extreme gains in efficiency [92]. Many of today's

applications that we find in data centers (e.g., machine learning applications [93] [94], signal processing applications [95], data analytics applications [96], and so forth) can effectively manage a certain degree of approximation without running into completely wrong computations. For example, for a k-means clustering algorithm, allowing a classification accuracy loss of 5% can save up to 50x energy [97]. Without going into too much detail, the main approximation techniques that we find in the literature are:

- *Custom data format*: creates smaller components by changing the precision (bit-width) of input or intermediate data to save both leakage and dynamic power [98];
- *Custom operator*: generates approximate adders and multipliers to perform partial computation with effects similar to those obtainable with *custom data format* [99] [100] [101];
- *Loop perforation*: skips some iterations of a loop to reduce computational overhead and so dynamic power [102];
- *Memoization*: stores the results of functions for later reuse to skip the portion of code in the second run [103]. This technique reduces the dynamic power but increases the leakage one. It requires some balance to be applied with confidence;
- *Task skipping*: skips memory references, tasks, or input portions to achieve energy efficiency [104]. Similar idea and effects as *loop perforation*;
- *Data sampling*: samples data from the input queue to speed up the execution [105]. Similar idea and effects as *loop perforation*;
- *Custom memory hierarchy*: adds more memory layers to hide the cache miss latency [106]. It reduces the leakage power due to data transmission over long distances, but by introducing new logic, it still increases the overall consumption. Also, in this case, the technique must be carefully balanced to obtain advantages;
- *Multiple inexact program version*: utilizes multiple versions of application code with different trade-offs between accuracy and overheads (e.g., execution time, power consumption) and selects at run time which one is better to use [105];

Applying AC is not always easy and requires wisely choosing the portion of code where to intervene and the technique not to have an unacceptable loss of quality. Further, careful monitoring of the output is required to ensure that quality specifications are met. New metrics are needed to estimate the error we are introducing when we use multiple techniques together (e.g., Monte Carlo simulation [107]). The work presented by Nepal et al. [108] combines *application logic level* and RTL techniques in a framework called ABACUS. ABACUS starts by creating an abstract syntax tree (AST) from the RTL description, and after it applies AC functions (e.g., custom data type, custom operator, and loop perforation) to create fair approximate designs. At last, it identifies the most suitable design along the Pareto frontier that represents the trade-off between accuracy and power consumption by exploring the space with all the possible variants. Only variants with a specific quality-of-service (QoS) are considered acceptable. Developing frameworks that lower the difficulty and the knowledge necessary to optimize power consumption is fundamental for the sector's evolution. Other research groups such as Chandrasekaran and Amira [91], Segal et al. [109], and Gao et al. [110] presented frameworks capable of implementing approximation techniques and estimating their effectiveness from the point of view of performance and energy consumption. Specifically, Gao and Qu suggest a runtime framework to exploit runtime energy information. The basic idea is to use a low-cost method like the error-resilient characteristics of each operator to estimate the impact of immediate input values on the accuracy of computation and then decide whether directly use the approximated value or perform an accurate computation. In general, AC is a powerful technique that allows saving from 40% up to 80% of the power consumption. However, if the data to be processed and the accelerator does not share the same data format, most of these benefits disappear, and some conversion is necessary [111]. In our opinion, conversion between data formats is an expensive operation from both an energy and time point of view; and it represents one of the biggest challenges within data centers where the vast heterogeneity of data often requires conversions. Table V summarise the optimization techniques and the main works presented in this subsection.

OVERALL IMPACTS CONSIDERATION: *Approximate computing reduces energy consumption by approximating the logic behind the processing algorithms. This approach impacts storage, terminal devices, and environmental ones. Many approximation techniques affect data and how it is read from memory. We remind you that the consumption of the memories depends on the amount of memory installed and the number of reads and write accesses made in the unit of time. Custom data format, memoization, and custom memory hierarchy are examples of optimizations that change the amount of memory used. Custom data format reduces memory usage by simplifying the data format to be more compact in memory. The technique is, therefore, also effective in reducing consumption by the storage component. On the contrary, memoization and custom memory hierarchy increase the amount of memory required and negatively affect the storage component of the data center. Optimizations such as data sampling, on the other hand, act on the number of accesses in the memory, guaranteeing higher energy savings, similar to the case just discussed with the custom data format. In conclusion, we consider the technique advantageous and applicable to data centers, especially when the main target is reinforcement learning or deep learning applications, where these techniques are almost mandatory. As mentioned in Section IV, multiple inexact program versions can be combined with innovative metrics to monitor and intervene quickly and autonomously on the accelerator, keeping its efficiency under control.*

D. Deployment level

This subsection discusses the dominant data center boards currently on the market and how data centers installed them. It represents an excellent guide to the choice of the product

TABLE V: Application logic Level - Power optimization techniques summary

Techniques	Main works	Target power	
		Dynamic	Leakage
Approximate computing	[91] [110] [111] [93] [99] [100] [105] [92] [108] [95] [109] [101]	✓	✗

and its installation in the data center, covering fundamental aspects given by the *Code of Conduct*, such as points ❸, ❹, and ❺. A data center is normally structured in areas called clusters. Racks are the base component of each cluster, and they can house up to 42 servers. Each rack is powered by a redundant PDU placed at the bottom of the structure and by at least one network switch for accessing the primary network. A secondary network often exists alongside the primary network, which gives direct access to other racks in the cluster or between the clusters themselves. Servers integrate FPGAs with PCI-e connection.

Currently, no cloud provider creates its FPGAs. Hence, the smallest unit of differentiation is the FPGA board both **Off-the-shelf** and **Custom** are possible [112]. The possible economic advantage depends on the quantities and the type of use. Table VI collects information on eleven off-the-shelf products between AMD Xilinx and Intel Altera. Each one is a general-purpose board with specific features (e.g., digital signal processing (DSP), random access memory (RAM), and HBM). The technologies with access to HBMs are the most promising for the HPC field since they have a high throughput at low power consumption. In recent years, Xilinx has created the product line Alveo specific for the data center environment. However, we do not find such a clear distinction for Altera products where the Stratix 10 is probably the best solution. Altera currently produces its FPGAs with a more advanced lithographic technology than Xilinx, although on paper, Xilinx appears to perform better in terms of power consumption with a ratio of 0.025 Watt/DSP for the Alveo U280 versus a ratio of 0.048 Watt/DSP for the FALCON Stratix 10. Figure 6 shows the energy consumption normalized on the number of DSPs embedded in the board. On the other hand, with custom FPGA boards, any feature can be varied, such as form factor, cooling, memory type and size, and FPGA family. This customization ensures that the boards closely match the requirements of the target system, which in many cases are stringent. We rarely encounter data center platforms designed natively for FPGAs; instead, we easily find upgrades of existing architectures with specific constraints on power supply, temperatures, form factor, and cooling system. Furthermore, in the custom solutions, the PMU is accessible from the CPU, allowing physical level optimization and better monitoring of the accelerator that perfectly matches the point ❹. For these systems, it is difficult, if not impossible, to find such specific boards on the market, and therefore, the design of a custom board is the only solution [113] [114]. For this reason, it would be appropriate to move towards boards that can be assembled through modules, thus reducing the need to develop custom boards.

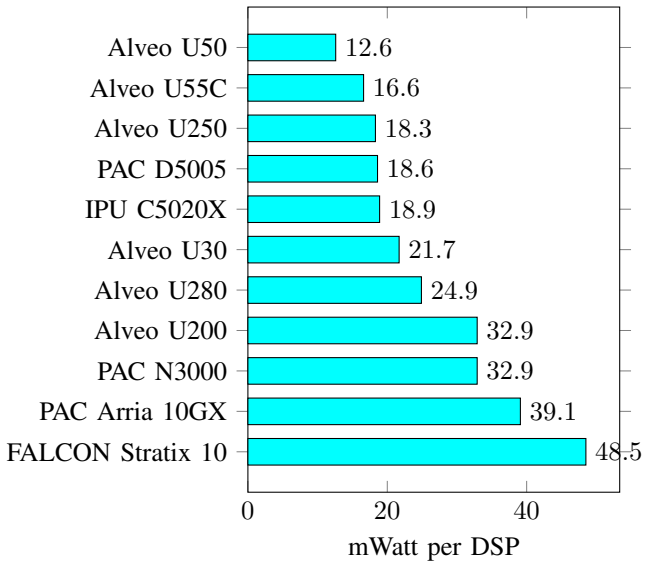


Fig. 6: Power consumption trends of the main off-the-shelf FPGA Data center boards.

TABLE VI: Main off-the-shelf FPGA Data center boards specification

Products	Technology	DSP	RAM	HBM
AMD Xilinx				
Alveo U30	16 nm	3456	8 GB	-
Alveo U50	16 nm	5952	-	8 GB
Alveo U55C	16 nm	9024	16 GB	-
Alveo U200	16 nm	6840	64 GB	-
Alveo U250	16 nm	12288	64 GB	-
Alveo U280	16 nm	9024	32 GB	8 GB
Intel Altera				
PAC Arria 10GX	20 nm	1687	8 GB	-
IPU C5020X	14 nm	3960	16 GB	-
PAC N3000	14 nm	3036	9 GB	-
FALCON Stratix 10	14 nm	3960	12 GB	8 GB
PAC D5005	14 nm	11520	32 GB	-

Integrating FPGA technology into a platform does not automatically mean saving energy and having an advantage. It all depends on the type of interconnectivity the FPGA has with the other elements inside the data center and on the actual utilization. If the FPGA does not exploit its resources, what you get is simply a system with higher losses due to the introduction of new hardware [115]. Bobda et al. [112] provide a complete analysis of the future of FPGA acceleration in data centers, describing in detail the currently most used placement techniques, but do not investigate the energy impacts that these choices entail. We can therefore identify two types of placement: **distributed** and **centralized**. Having a distributed FPGA placement means that nodes have their FPGAs, and other nodes can access it. This solution leads to greater utilization, more scalability, and more power reduction since the CPU workload is offloaded more effectively to an

FPGA pool [8] [116]. It is also possible to place FPGAs in a centralized manner. Typically we see this choice for network architectures, where the single FPGA is coupled to an ASIC or a CPU to implement a smart network interface card (NIC) or Smart Switch. Such a deployment needs fewer FPGAs; this typically translates to easier management, the lower total cost of ownership (TCO), and smaller average node sizes [112]. We strongly recommend the adoption of FPGAs in network devices. Due to their structure and composition, the implementation of network protocols on FPGAs is advantageous with very high performance. Furthermore, this allows the creation of custom network protocols with a significant impact on data analysis applications, or in general, on applications where most of the time is spent transferring data from storage to computing nodes.

Virtually all data centers today implement FPGAs in a distributed way because it ensures the always exploitation of the resources and that the idle moments of the FPGAs are at a minimum [117]. If a node does not use its local FPGA, another node can take possession of it and use it to speed up its workflow. Implementing a distributed infrastructure involves many challenges:

1) the power consumption due to the introduction of many FPGAs must not exceed the amount of power saved by their use;
2) the internal data center network infrastructure must be redesigned to allow access to FPGAs between nodes, increasing other sources of consumption;
3) the management and monitoring are more complex, and there can be concurrency problems;

Several papers [118] [119] [120] [121] [122] present examples of architecture of this type. However, in almost all systems, the network equipment necessary for the communication between FPGAs is implemented internally to the FPGAs, consuming resources. Developing and using new switches capable of supporting FPGAs would be a great help to the increasingly easy integration of FPGAs [123] [124]. More recent lines of research focus on the third point, the management of this new resource within the data center [125] and facilitating its development [126]. The most famous platforms, such as Openstack [127] and Kubernetes [128], which instantiate, and reconfigure data center resources in real-time, can effectively optimize energy consumption but do not have support for FPGAs. A possible new field of research could be to extend these tools by understanding what it means to optimize FPGAs, together with the other components already present. Table VII summarise the optimization techniques and the main works presented in this subsection.

TABLE VII: Deployment Level - Power optimization techniques summary

Techniques	Main works	Target power	
		Dynamic	Leakage
Off-the-shelf board	[118] [119] [120] [121] [122]	✗	✓
Custom board	[8] [113] [114] [116]	✓	✓
Distributed placement	[8] [118] [119] [113] [120] [114] [116] [121] [122]	✗	✓

OVERALL IMPACTS CONSIDERATION: *At this level, the data center architecture is defined, and the choices made will impact each source of consumption presented. Each board has a different consumption and amount of memory as presented in Table VI and Figure 6. In addition to the memory typology, there is a different kind of cooling system and network interconnection installed on the board. These features affect the consumption of the terminals, environmental, and storage devices. Custom solutions allow the designer to better balance these aspects by producing boards specific to the type of architecture designed. Of course, it is not always possible to opt for solutions of this type due to the development costs incurred. Also, it is essential to consider what types of workloads the data center will be computing to understand whether to adopt FPGAs in the computing nodes, in the network components, or both. Many applications have benefits in speeding up communications, and some of the computations, such as data filtering, can already be applied at the network level while also reducing the resulting traffic. The most commonly accepted choice of implementing FPGAs in a distributed way hurts consumption related to the network that must allow FPGAs to communicate with each other. The type of network implemented and the number of networks created define the overall impact. From the literature analyzed, we discover that the most common type of network is the torus and that a maximum of two networks are created [8] [113] [116]. The first network, where the data center exchanges the data in and out, is called primary. The second is called secondary, which connects the individual FPGAs, allowing the sharing of resources. The increase in network consumption that we obtain is, therefore, a side effect of this solution which can be considered acceptable given the advantages of flexibility and use that it guarantees. We believe that particular attention should be paid to these decisions as they impact the entire structure of the data center and its final performance. The last observation we want to report concerns the type of FPGA adopted. Small FPGAs, in terms of resources, have faster development times. In about an hour, the hardware designer has the final configuration file, while for large FPGAs, such as the Alveo U280, times can go up to 15 hours. So negatively affects the overall development times. On the other hand, smaller FPGAs have lower performance than large ones, but a good network infrastructure and wise use of FPGAs in a distributed manner can compensate. In the end, we recommend using the latter solution when considering FPGA integration in data centers.*

E. Infrastructure level

The infrastructure level represents the highest level of abstraction that we have identified to which it is possible to apply energy optimizations that include the use of FPGAs. The techniques presented in this subsection apply to large data center areas such as clusters because they require a comprehensive understanding of the current workflow. The main idea at this level is to reduce the fragmentation of

resources, thus increasing their utilization. This idea shrinks execution to fewer servers, and it is possible to shut down everything else to save energy. We are therefore talking about techniques such as:

- FPGA resource virtualization: exposes the internal resources of the FPGA as if they were virtualized, favoring their full utilization;
- Accelerated virtual machine (AVM) placement: determines where to allocate the virtual machines (VMs) according to the type of workflow and energy constraints;
- Accelerated Job scheduling: improves the utilization of cloud servers equipped with FPGAs;
- Reconfiguration: reconfigures FPGAs at run time according to the state of consumption of the data center;

FPGA resource virtualization has always been an ambitious idea ever since the early days of partial reconfiguration [129]. Popular virtualization techniques used in the software do not apply to FPGAs since they do not execute sequential programs but implement parallel circuits [130]. Guo et al. [131] identify several techniques like *slot-based allocation* [129], *FPGA overlays* [132], and *standalone resource sharing* [133]. The first divides the FPGA into fixed areas that accommodate pre-defined circuits and take advantage of partial reconfiguration at run time to change the type of behavior. This architecture adds a layer of complexity to the management of FPGA resources. Examples of this, extended to data center solution, could be [134] [135] [136]. Bobda et al. [112] collect many works on the subject in their survey, finding that the overhead on the consumption of resources by applying this technique does not exceed 30%. [113] made similar reasoning, where partial reconfiguration is presented as one of the most important features that FPGAs offers. They propose standards for the input/output interface to freely reconfigure the kernel without ceasing to read the FPGA input stream. The second technique instead exposes a simplified layer that increases the programmability and productivity of FPGAs [137]. With this type of approach, it is possible to treat FPGAs similarly to CPUs, making it more convenient for traditional virtualization [138]. In standalone resource sharing, the hardware resources on the FPGA board can be categorized based on programmability, identifying the types of resources present and classifying them into logic, connectivity, or memory resources [133]. These can then be shared individually by generating three sharing models. Respectively we will have: configuration, bandwidth, and capacity sharing. Each of these models has different properties. Configuration sharing concerns sharing logic between multiple FPGAs to speed up the computation. Bandwidth sharing, on the other hand, aims to share network or memory connectivity to improve the movement of large amounts of data. Finally, capacity sharing goes to sharing the memory on the board.

FPGA resource virtualization, therefore, try to increase the utilization of the FPGA by exposing its resources to more processes, thus avoiding having to use other accelerators and saving energy. Of course, sharing FPGAs expose the cloud to security issues. Different types of attacks and countermeasures have been proposed over the years. They classify cloud FPGA attacks into: *extraction attacks*, with the idea to extract information from observable quantities such as power consumption and frequency [139] [140], *fault injection attacks* with the idea to introduce glitches in the hardware in an attempt to stop the accelerator or extract information [141] and *denial of Service attacks* with the idea to disable the use of the accelerator, for example, by invalidating the shared DRAM with a row hammers [142] attack. For a discussion on these attacks, we remained the reader to [143], a complete survey on the topic. Fortunately, these attacks are known to the research community, and several countermeasures have already been implemented. Amazon AWS, for example, implements a bitstream antivirus [143] that scans the sent bitstream for harmful structures before programming the FPGA. Also, it only allows the loading of bitstreams generated with its execution stream. Additionally, most recent FPGAs, such as those from AMD Xilinx, include isolated partial reconfiguration capabilities that allow for independent partitioning and scheduling of FPGA resources and configure secure access [144]. However, like every security countermeasure, these solutions negatively impact the power consumption of the data center.

As we have often pointed out, the high power consumption of accelerators, along with their under-utilization, can impose high operating costs [145]. This sentence explains why virtualization is so relevant. **AVM placement** problem is well known and discussed in the literature, but not many papers extend the problem by considering FPGAs. Yarahmadi et al. [146] introduce a genetic algorithm-based VM placement method capable of responding to these needs. The goal is to minimize the postponement of the requests while staying energy-efficient and compensate for the enormous exploration space created by limiting the number of chromosomes as a function of the number of VMs. Zhang et al. [147] present a similar work with the difference that, in this case, they provide an advanced energy model of the data center.

Correlated to the placement problem, we always find **Accelerated job scheduling** problems. Scheduling on FPGA systems can be considered a classical resource-constrained scheduling problem (RCSP). We know the classical RCSP problem as AVM placement is a strongly NP-hard problem. Thus, the FPGA community dedicates much attention to the design of heuristics [148]. Many research groups have approached this problem under different aspects [135] [149] [150]. Dhar et al. [151] propose a methodology for scheduling heterogeneous tasks across an FPGA considering the possibility of partial reconfiguration and sharing of resources. This methodology reduces the resource fragmentation and can optimize spaces and leakage power. Ting Loke and Yang Koay [152] instead approach the problem from an innovative point of view. Their idea consists in scheduling the tasks considering the relative deadline for each one. If a task has a lot of time available, they increase its latency by applying DFS optimizations and clock gating at run time. In this way, they obtain an adaptive model that regulates energy optimization according to the time available and the type of task.

We cannot conclude our discussion without talking about the main functionality that FPGAs offer; the **reconfiguration**.

TABLE VIII: Infrastructure Level - Power optimization techniques summary

Techniques	Main works	Target power	
		Dynamic	Leakage
FPGA resource virtualization	[134] [129] [137] [112] [135] [132] [153] [136]	✗	✓
Accelerated virtual machine	[146] [147]	✗	✓
Accelerated job scheduling	[148] [135] [151] [149] [152] [150] [154]	✓	✓
Reconfiguration	[8] [113]	✓	✓

The ability to reconfigure the hardware allows the data center to implement different energy policies during computation and to test optimizations and solutions that would otherwise be impossible without updating the entire hardware. This feature is not free, and we must consider the times introduced by the process. Updating the bitstream of a single FPGA can take several seconds. Organizing the data center with a distributed FPGA placement and the use of FPGA overlays greatly reduces this overhead, guaranteeing rapid updating of entire clusters contemporary. Table VIII summarise the optimization techniques and the main works presented in this subsection.

OVERALL IMPACTS CONSIDERATION: *The techniques presented in this subsection aim to maximize the use of FPGAs in the data center to enhance investment. This procedure increases the energy consumption related to the individual FPGAs, even if limited, to reduce the workload on the more traditional CPU-based computing nodes and, therefore, their consumption. FPGA resource virtualization harms network consumption but positively on storage consumption, allowing to share of memory resources between FPGAs across the network and maximizing their use. We can do the same reasoning with accelerated virtual machine placement, where the aim is to increase the utilization of the single node and avoid having to switch on more nodes to manage a specific workload. Finally, we discuss the fundamental feature of FPGAs reconfiguration. The reconfiguration acts on every aspect of the data center, allowing the deployment of FPGAs into servers, network devices, and control systems. The reconfiguration keeps the data center updated for future challenges by guaranteeing flexibility and maintainability.* We consider virtualization or resource-sharing techniques on FPGAs to be vital for the development of efficient and sustainable data centers.

VI. Power Optimization Trends

Data center systems are subject to constraints such as TCO, PUE, performance, resilience, modularity, and scalability that reduce the degree of freedom on the possibility. From the Bobda et al. [112] analysis, all service providers prefer to have FPGAs in a distributed architecture that guarantees maximum flexibility and scalability to the system. Integrating off-the-shelf products such as existing data center upgrades is rarely possible. The form factor, the cooling system, and the constraints on the consumption of the single server are the main reasons why this integration does not take place. This problem leads to investing in custom solutions, which, in addition to being specific for a single environment, are also better designed from an energy point of view, allowing the board voltages to be monitored and managed more precisely.

Optimization techniques widely used today are power gating, approximate computing, AVM placement, and accelerated job scheduling. The reasons are many. Power gating is a simple technique that does not require calibration systems and does not introduce further problems as it does for DVS. Since FPGAs do not have specific hardware for floating-point computation, optimizations on the type of data used can almost always be found by preferring solutions where the calculation is in fixed-point. This practice is common in data centers where the primary workload is statistical data analysis. Machine learning algorithms are resilient to data approximation techniques, guaranteeing the same results. AVM placement and accelerated job scheduling are extensions of the already present and widely used VM management algorithms in the data center. These techniques provide great flexibility in managing the workload and servers available, which is essential for the performance, resilience, and modularity that a data center must have.

VII. Potential Future Innovations

We identify areas of potential novelty by considering the optimization techniques used today and comparing them with the needs presented in the woks summarized into Tables III, IV, V, VII, and VIII.

Power models: The most commonly used energy models do not consider the device temperature. Works such as the one presented by Khaleghi et al. [90] show us how seeing the thermal profile of the chip allows more studied optimizations. The Thermodynamics computing model [32] is an excellent replacement for today's model. Unifying the entropy of Shannon with the entropy in thermodynamics and with the metrics of Kolmogorov would allow us to estimate the energy and thermal consumptions from the drafting of the algorithm alone without having to simulate the obtained solution from time to time. Some work exists in the literature but is still very limited and hard to use in real applications [155]. This model would then find a particular application in FPGAs where high-level code directly impacts the generation of the hardware, allowing optimization of energy and temperature of the chip contemporary. Also, there is a need for new metrics that consider the type of energy consumed, whether it comes from renewable sources or fossil sources. Section 3 proposes an example, but the search field is still open. Finally, data center providers could consider using energy consumption as a unit of measurement for the utilization of their resources as they already do with time units. So, for example, setting a consumption limit as a constraint for the user.

Type of boards: As discussed in Section VI, the numerous data center constraints prevent the direct integration of

off-the-shelf products. A fundamental step in the diffusion of FPGAs in this sector is the development of modular boards that can be assembled according to specific needs [112], thus reducing the costs of experimenting and designing custom solutions. The possibility of adding exact control modules for energy would allow monitoring of the consumption of individual servers with greater granularity and smoothly implement optimization techniques on a physical level. This is also what the data center providers need to respect the code of conduct (points ❸, ❹, and ❺).

Adaptive systems: Introducing new hardware into an architecture leads to management challenges. It is not always possible to individually optimize each accelerator for a specific workload. Adaptive systems on FPGAs able to adopt the energy profile best suited to a given workflow would greatly help to manage this technology and its diffusion. Systems of this type could strongly limit energy consumption by identifying invisible configurations at design time, even going so far as to self-reconfigure to work better.

Management tools: The most famous platforms, such as Openstack [127] and Kubernetes [128], which instantiate, and reconfigure data center resources in real-time, can effectively optimize energy consumption but do not have support for FPGAs. Extending these tools with FPGA support is, of course, a way to ensure the continued use of FPGAs in data centers even in the long term. In particular, the new possibility should be to virtualize these new resources, monitor their consumption and possible configuration errors (as appears for Microsoft's Catapult II project [113]), and easily reconfigure them remotely. All act with different granularity at the single server, rack, or cluster level.

A. Green data distribution and challenges

Recently Google introduced the concept of **Carbon-Aware Computing** [156], which exploits flexibility in when, where, and how computing takes place to reduce carbon emissions. The idea is not new. Some computing has flexibility in when it can run, like processing videos, feature extraction and training large-scale machine learning models, simulation pipelines, and many other latency-tolerant workloads. So, they shape the load profile over 24 hours to balance the absorbed pick energy. Some computing jobs have flexibility in where they can run, like user-facing services that can be geographically rebalanced or resourced (common for Microsoft's applications, Facebook, and Twitter messaging applications) to a greener data center like the one with FPGA support. Today there are two deployed systems Microsoft's Carbon Aware Kubernetes [157] and Google's Carbon-Intelligent computing platform [158]. The challenges, in this case, consist of understanding which types of workloads enjoy the properties presented, ensuring that spatially flexible load ends up in the right location, and how load shaping affects CO_2 emission.

VIII. Conclusion

The article exposes today's state-of-the-art energy optimization techniques involving FPGAs in data centers. These techniques improve and enhance the benefit given by FPGAs with a view to sustainable computation. The article focuses on metrics, monitoring power consumption models, existing FPGAs power optimization techniques, and open challenges we must achieve in energy efficiency. Data center designers must understand this information well to prioritize energy consumption during design. Optimization techniques widely used today are power gating, approximate computing, FPGA resource virtualization, AVM placement, and accelerated job scheduling. These techniques push the utilization of FPGAs to the maximum, their performance, and their energy efficiency while also giving great flexibility in managing the workload and servers available, which is essential for the performance, resilience, and modularity that a data center must-have. We believe that FPGAs, with the proper observations and thanks to their low latency, high throughput, and energy efficiency, can be a complementary alternative to CPUs and GPUs. Especially, when the principal workloads are statistical data analysis, AI, computer vision, and security. The evolution of this technology, in the short term, will determine its long-term use in data centers. Many obstacles are still present. The main one is the lack of support for FPGAs in the asset management and data center monitoring tools. Besides, there is no comprehensive software stack to deploy them on the cloud. The extension of software such as OpenStack and Kubernetes to FPGAs is what cloud service providers are looking for to consider FPGAs an attractive product. Much has been done and will be done in the future to make FPGAs more attractive. Just think that FPGAs was born to facilitate the design of ASICs, and today we find them in the largest data centers in the world. In the future, we will probably see the birth of two types of products, one focused on performance and one focused on consumption and therefore to greater diversity and applicability of this technology, for sure if we will consider the need for modular and easily customizable boards.

Acknowledgments

This work was supported by the Italian Ministry of University and Research (MUR) and the European Union (EU) under the PON/REACT project and by the Horizon 2020 EU Research & Innovation Programme under grant agreement No. 957269 (EVEREST project).

References

[1] S. R. Department. (2022) Amount of data created, consumed, and stored 2010-2025. [Online]. Available: https://www.statista.com/statistics/871513/worldwide-data-created/#:~:text=The\%20total\%20amount\%20of\%20data,to\%20more\%20than\%20180\%20zettabytes.

[2] C. Vinay, H. Vikas, G. Vatsal, and G. Mohsen, "A comprehensive review of the covid-19 pandemic and the role of iot, drones, ai, blockchain, and 5g in managing its impact," *IEEE Access*, vol. 8, pp. 90 225 – 90 265, May 2020.

[3] T. Bhattacharya and X. Qin, "Modeling energy efficiency of future green data centers," in *2020 11th International Green and Sustainable Computing Workshops (IGSC)*, 2020, pp. 1–3.

[4] X. Peng and X. Qin, "Energy efficient data centers powered by on-site renewable energy and ups devices," in *2020 11th International Green and Sustainable Computing Workshops (IGSC)*, 2020, pp. 1–3.

[5] I. René, H. Roland, and et al., "Digital transformation—life cycle assessment of digital services, multifunctional devices and cloud computing," in *The International Journal of Life Cycle Assessment volume*, 2020.

[6] M. Wiboonrat, "Data centre optimization for energy efficiency improvement," in *2014 International Conference on Information Science, Electronics and Electrical Engineering*, vol. 3, 2014, pp. 1779–1787.

[7] S. R. Department. (2022) Energy demand in data centers worldwide from 2015 to 2021, by type. [Online]. Available: https://www.statista.com/statistics/186992/global-derived-electricity-consumption-in-data-centers-and-telecoms/

[8] P. A., C. A. M., C. E. S., C. D., C. K., D. J., E. H., F. J., G. G. P., G. J., H. M., H. S., H. S., H. A., K. J., L. S., L. J., P. E., P. S., S. A., T. J., X. P. Y., and B. D., "A reconfigurable fabric for accelerating large-scale data center services." in *ACM/IEEE 41st International Symposium on Computer Architecture (ISCA).*, 2014, pp. 13–24.

[9] G. global market insights. (2022) Fpga market in the last year. [Online]. Available: https://www.gminsights.com/industry-analysis/field-programmable-gate-array-fpga-market-size#:~:text=Industry\%20Trends,computing\%20applications\%20in\%20data\%20centers.

[10] C. Pilato, S. Bohm, F. Brocheton, J. Castrillon, R. Cevasco, and et al., "EVEREST: A design environment for extreme-scale big data analytics on heterogeneous platforms," in *IEEE/EDAA Design, Automation & Test in Europe Conference (DATE)*, 2021, pp. 1–6.

[11] M. Qasaimeh, J. Zambreno, P. H. Jones, K. Denolf, J. Lo, and K. Vissers, "Analyzing the energy-efficiency of vision kernels on embedded cpu, gpu and fpga platforms," in *2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, 2019, pp. 336–336.

[12] J. S.-L. Caspar Honée, Daniel Hedin and M. Fröling, "Environmental performance of data centres - a case study of the swedish national insurance administration." in *2012 Electronics Goes Green 2012+*, 2012. [Online]. Available: https://ieeexplore.ieee.org/document/6360435

[13] X. Chang, S. Yang, Y. Jiang, X. Xie, and X. Tang, "Research on key energy-saving technologies in green data centers," in *2020 IEEE International Conference on Smart Cloud (SmartCloud)*, 2020, pp. 111–115.

[14] J. Wan, J. Zhou, and X. Gui, "Sustainability analysis of green data centers with cchp and waste heat reuse systems," *IEEE Transactions on Sustainable Computing*, vol. 6, no. 1, pp. 155–167, 2021.

[15] M. Dayarathna, Y. Wen, and R. Fan, "Data center energy consumption modeling: A survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 732–794, 2016.

[16] H. Svein, Atle and V. Ole, Sten. (2022) Green mountain data center. [Online]. Available: https://greenmountain.no/

[17] P. B. Maria Avgerinou and L. Castellazzi, "Trends in data centre energy consumption under the european code of conduct for data centre energy efficiency." in *ENERGIES*, 2017, pp. 147–1480. [Online]. Available: https://doi.org/10.3390/en10101470

[18] A. Xilinx. (2011) Amd xilinx fpga: Spartan 6. [Online]. Available: https://www.xilinx.com/products/silicon-devices/fpga/spartan-6.html

[19] I. Altera. (2019) Intel altera fpga: Agilex. [Online]. Available: https://www.intel.it/content/www/it/it/products/details/fpga/agilex.html

[20] Intel. (2022) Fpga technology at crossroads. [Online]. Available: https://www.crossroadsfpga.org/

[21] H. Jun, J. Cho, K. Lee, H.-Y. Son, K. Kim, H. Jin, and K. Kim, "Hbm (high bandwidth memory) dram technology and architecture," in *IEEE International Memory Workshop (IMW)*, 2017, pp. 1–4.

[22] Y. Choi, Y. Chi, J. Wang, L. Guo, and J. Cong, "When HLS meets FPGA HBM: benchmarking and bandwidth optimization," *CoRR*, vol. abs/2010.06075, 2020. [Online]. Available: https://arxiv.org/abs/2010.06075

[23] S. Soldavini, K. F. A. Friebel, M. Tibaldi, G. Hempel, J. Castrillon, and C. Pilato, "Automatic creation of high-bandwidth memory architectures from domain-specific languages: The case of computational fluid dynamics," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 2022.

[24] S. Soldavini, D. Sciuto, and C. Pilato, "Iris: Automatic generation of efficient data layouts for high bandwidth utilization," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2023.

[25] Intel. (2019) Understanding how the new intel hyperflex fpga architecture enables next-generation high-performance systems. [Online]. Available: https://www.intel.com/content/dam/support/us/en/programmable/support-resources/bulk-container/pdfs/literature/wp/wp-01231-understanding-how-hyperflex-architecture-enables-high-performance-syste pdf

[26] R. Nane, V.-M. Sima, C. Pilato, J. Choi, B. Fort, A. Canis, Y. T. Chen, H. Hsiao, S. Brown, F. Ferrandi, J. Anderson, and K. Bertels, "A survey and evaluation of fpga high-level synthesis tools," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 10, pp. 1591–1604, 2016.

[27] F. Ferrandi, V. G. Castellana, S. Curzel, and et al., "Invited: Bambu: an open-source research framework for the high-level synthesis of complex applications," in *ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 1327–1330.

[28] J. Cong, M. Huang, P. Pan, Y. Wang, and P. Zhang, "Source-to-source optimization for hls." in *FPGAs for Software Programmers*, D. Koch, F. Hannig, and D. Ziener, Eds. Springer, 2016, pp. 137–163.

[29] S. Atefeh, Y. Cody, Hao, G. Min, and C. Jason, "Autodse: Enabling software programmers to design efficient fpga accelerators," in *ACM Transactions on Design Automation of Electronic Systems*, vol. 27, 2022, p. 32.

[30] A. Xilinx. (2021) Xilinx vitis hls llvm 2021.2. [Online]. Available: https://github.com/Xilinx/HLS

[31] W. Wu, H. Chen, K. Li, and J. Yu, "Overview of typical application energy efficiency optimization in high-performance data centers," in *2021 IEEE International Conference on Power Electronics, Computer Applications (ICPECA)*, 2021, pp. 702–705.

[32] C. Tom, D. Erik, G. Natesh, and e. a. Todd, "Thermodynamic computing," in *A Computing Community Consortium (CCC) workshop report*, 2019, p. 36.

[33] Q. Zhang, Z. Meng, X. Hong, Y. Zhan, J. Liu, J. Dong, T. Bai, J. Niu, and M. J. Deen, "A survey on data center cooling systems: Technology, power consumption modeling and control strategy optimization," *Journal of Systems Architecture*, vol. 119, p. 102253, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383762121001739

[34] Google. (2021) Announcing new tools to measure and reduce your environmental impact. [Online]. Available: https://cloud.google.com/blog/topics/sustainability/new-tools-to-measure-and-reduce-your-environmental-impact

[35] S. DIOUANI and H. MEDROMI, "How energy consumption in the cloud data center is calculated," in *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*, 2019, pp. 1–10.

[36] V. M. Fthenakis and H. C. Kim, "Greenhouse-gas emissions from solar electric- and nuclear power: A life-cycle study." in *Energy Policy.*, 2006, pp. 2549–2557. [Online]. Available: https://doi.org/10.1016/j.enpol.2006.06.022

[37] S. Pacca and A. Horvath, "Greenhouse gas emissions from building and operating electric power plants in the upper colorado river basin." in *Environ. Sci. Technol.*, 2002, pp. 3194–3200. [Online]. Available: https://doi.org/10.1021/es0155884

[38] Unece. (2021) Life cycle assessment of electricity generation options. [Online]. Available: https://unece.org/sites/default/files/2021-10/LCA-2.pdf

[39] Y. C. Amip J. Shah and C. E. Bash, "Sources of variability in data center lifecycle assessment." in *2012 IEEE International Symposium on Sustainable Systems and Technology (ISSST)*, 2012. [Online]. Available: https://doi.org/10.1109/ISSST.2012.6227975

[40] D. Beth and A. S., "The life cycle assessment of a uk data centre," in *Int J Life Cycle Assess*, 2015, p. 332–349. [Online]. Available: https://doi.org/10.1007/s11367-014-0838-7

[41] V. K., M. G., J. P., and C. F., "Modeling the temperature bias of power consumption for nanometer-scale cpus in application processors," in *International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIV)*, 2014.

[42] T. K., B. P., and T. D., "Voltage scaling based green design on ultra scale fpga," in *2013 International Conference on Green Computing, Communication and Conservation of Energy (ICGCE)*, 2013, pp. 125–129.

[43] J. Nunez-Yanez, V. Chouliaras, and J. Gaisler, "Dynamic voltage scaling in a fpga-based system-on-chip," in *2007 International Conference on Field Programmable Logic and Applications*, 2007, pp. 459–462.

[44] C. Chow, L. Tsui, P. Leong, W. Luk, and S. Wilton, "Dynamic voltage scaling for commercial fpgas," in *Proceedings. 2005 IEEE International Conference on Field-Programmable Technology, 2005.*, 2005, pp. 173–180.

[45] H. Qi, O. Ayorinde, and B. H. C., "An energy-efficient near/sub-threshold fpga interconnect architecture using dynamic voltage scaling and power-gating," in *2016 International Conference on Field-Programmable Technology (FPT)*, 2016, pp. 20–27.

[46] I. Ahmed, S. Zhao, O. Trescases, and V. Betz, "Measure twice and cut once: Robust dynamic voltage scaling for fpgas," in *2016 26th International Conference on Field Programmable Logic and Applications (FPL)*, 2016, pp. 1–11.

[47] N. Atukem and Nunez-Yanez, "Adaptive voltage scaling in a dynamically reconfigurable fpga-based platform," in *ACM Trans. Reconfig. Technol. Syst.*, vol. 5, 2012, p. 22.

[48] J. L. Nunez-Yanez, "Adaptive voltage scaling with in-situ detectors in commercial fpgas," *IEEE Transactions on Computers*, vol. 64, no. 1, pp. 45–53, 2015.

[49] I. Ahmed, S. Zhao, J. Meijers, O. Trescases, and V. Betz, "Automatic bram testing for robust dynamic voltage scaling for fpgas," in *2018 28th International Conference on Field Programmable Logic and Applications (FPL)*, 2018, pp. 68–687.

[50] B. Pontikakis, H. T. Bui, F.-R. B., and Y. Savaria, "A low-complexity high-speed clock generator for dynamic frequency scaling of FPGA and standard-cell based designs," in *IEEE International Symposium on Circuits and Systems*, 2007, pp. 633–636.

[51] S. Zhao, I. Ahmed, C. Lamoureux, A. Lotfi, V. Betz, and O. Trescases, "A universal self-calibrating dynamic voltage and frequency scaling (dvfs) scheme with thermal compensation for energy savings in fpgas," in *2016 IEEE Applied Power Electronics Conference and Exposition (APEC)*, 2016, pp. 1882–1887.

[52] Z. Seifoori, B. Khaleghi, and H. Asadi, "A power gating switch box architecture in routing network of sram-based fpgas in dark silicon era," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, 2017, pp. 1342–1347.

[53] A. A. M. Bsoul and S. J. E. Wilton, "An fpga architecture supporting dynamically controlled power gating," in *2010 International Conference on Field-Programmable Technology*, 2010, pp. 1–8.

[54] A. Rahman, S. Das, T. Tuan, and S. Trimberger, "Determination of power gating granularity for fpga fabric," in *IEEE Custom Integrated Circuits Conference 2006*, 2006, pp. 9–12.

[55] S. Ishihara, M. Hariyama, and M. Kameyama, "A low-power fpga based on autonomous fine-grain power gating," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 8, pp. 1394–1406, 2011.

[56] H. T. and L. T. K., "Power gating technique in pacemaker design on fpga," in *The 2012 International Conference on Advanced Technologies for Communications*, 2012, pp. 14–18.

[57] T. Tuan, A. Rahman, S. Das, S. Trimberger, and S. Kao, "A 90-nm low-power fpga for battery-powered applications," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, pp. 296–300, 2007.

[58] M. Hosseinabady and J. L. Nunez-Yanez, "Energy optimization of fpga-based stream-oriented computing with power gating," in *2015 25th International Conference on Field Programmable Logic and Applications (FPL)*, 2015, pp. 1–6.

[59] H. Qi, O. Ayorinde, and B. H. Calhoun, "An ultra-low-power fpga for iot applications," in *2017 IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference (S3S)*, 2017, pp. 1–3.

[60] X. Tang, E. Giacomin, B. Chauviere, A. Alacchi, and P. Gaillardon, "Openfpga: An open-source framework for agile prototyping customizable fpgas," *IEEE Micro*, vol. 40, no. 04, pp. 41–48, jul 2020.

[61] J. Luis Nunez-Yanez, M. Hosseinabady, and A. Beldachi, "Energy optimization in commercial fpgas with voltage, frequency and logic scaling," *IEEE Transactions on Computers*, vol. 65, no. 5, pp. 1484–1493, 2016.

[62] A. Raghunathan, S. Dey, and N. Jha, "Register-transfer level estimation techniques for switching activity and power consumption," in *Proceedings of International Conference on Computer Aided Design*, 1996, pp. 158–165.

[63] E. Bezati, S. Casale-Brunet, M. M., and J. W. J., "Clock-gating of streaming applications for energy efficient implementations on fpgas," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 4, pp. 699–703, 2017.

[64] P. Georgios, "Clock gating methodologies and tools: a survey," in *International journal of Circuit theory and Applications*, 2015.

[65] S. Huda, M. Mallick, and J. H. Anderson, "Clock gating architectures for fpga power reduction," in *2009 International Conference on Field Programmable Logic and Applications*, 2009, pp. 112–118.

[66] Q. Wu, M. Pedram, and X. Wu, "Clock-gating and its application to low power design of sequential circuits," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 3, pp. 415–420, 2000.

[67] P. Bhaskar and V. K. Jathar, "Development of processor engine for fpga based clock gating and performing power analysis," in *2016 International Conference on Computing Communication Control and automation (ICCUBEA)*, 2016, pp. 1–5.

[68] J. P. Oliver, J. Curto, D. B., M. R., and E. B., "Clock gating and clock enable for fpga power reduction," in *2012 VIII Southern Conference on Programmable Logic*, 2012, pp. 1–5.

[69] Y. Z., J. Roivainen, and A. Mammela, "Clock-gating in fpgas: A novel and comparative evaluation," in *9th EUROMICRO Conference on Digital System Design (DSD'06)*, 2006, pp. '584–590.

[70] W. Osborne, W. Luk, J. Coutinho, and O. Mencer, "Reconfigurable design with clock gating," in *2008 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation*, 2008, pp. 187–194.

[71] B. Pandey, J. Yadav, M. Pattanaik, and N. Rajoria, "Clock gating based energy efficient alu design and implementation on fpga," in *2013 International Conference on Energy Efficient Technologies for Sustainability*, 2013, pp. 93–97.

[72] B.-L. Tan, W.-K. Lee, K.-M. Mok, and H.-G. Goh, "Clock gating implementation on commercial field programmable gate array (fpga)," in *2018 4th International Conference on Electrical, Electronics and System Engineering (ICEESE)*, 2018, pp. 102–106.

[73] B. Pandey, D. Singh, D. Baghel, J. Yadav, and M. Pattanaik, "Clock gated low power memory implementation on virtex-6 fpga," in *2013 5th International Conference and Computational Intelligence and Communication Networks*, 2013, pp. 409–412.

[74] L. Sterpone, L. Carro, D. Matos, S. Wong, and F. Fakhar, "A new reconfigurable clock-gating technique for low power sram-based fpgas," in *2011 Design, Automation Test in Europe*, 2011, pp. 1–6.

[75] T. Agrawal, A. Kumar, and S. K. Saraswat, "Design of low power sram on artix-7 fpga," in *2016 2nd International Conference of Communication Control and Intelligent Systems (CCIS)*, 2016, pp. 203–209.

[76] M. Geier, M. Brändle, and S. Chakraborty, "Insert amp; save: Energy optimization in ip core integration for fpga-based real-time systems," in *2021 IEEE 27th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2021, pp. 80–91.

[77] Y. Z., Q. T., L. Li, W. Wang, K. Choi, J. J., H. J., and S.-Y. A., "Automatic register transfer level cad tool design for advanced clock gating and low power schemes," in *International SoC Design Conference (ISOCC)*, 2012, pp. 21–24.

[78] Siemens. (2022) Powerpro power analysis and optimization platform. [Online]. Available: https://eda.sw.siemens.com/en-US/ic/powerpro

[79] V. George, H. Zhang, and J. Rabaey, "The design of a low energy fpga," in *Proceedings. 1999 International Symposium on Low Power Electronics and Design (Cat. No.99TH8477)*, 1999, pp. 188–193.

[80] A. Sharma, R. Ansar, M. S. Gaur, L. B., and V. Laxmi, "Reducing fifo buffer power using architectural alternatives at rtl," in *2016 20th International Symposium on VLSI Design and Test (VDAT)*, 2016, pp. 1–2.

[81] A. Garcia, L. Perez, and R. Acuna, "Power consumption management on fpga," in *15th International Conference on Electronics, Communications and Computers (CONIELECOMP'05)*, 2005, pp. 240–245.

[82] M. Zamani and E. Esmaili, "Reducing power consumption in fpga routing," in *CCECE 2003 - Canadian Conference on Electrical and Computer Engineering. Toward a Caring and Humane Technology (Cat. No.03CH37436)*, vol. 1, 2003, pp. 9–12 vol.1.

[83] B. V., R. J., and M. A., "Architecture and cad for deep-submicron fpgas," in *Kluwer Academic Publishers*, 1999.

[84] C. Hoo, Y. Ha, and A. Kumar, "A directional coarse-grained power gated fpga switch box and power gating aware routing algorithm," in *2013 23rd International Conference on Field programmable Logic and Applications*, 2013, pp. 1–4.

[85] G. V. Leming and K. Nepal, "Low-power fpga routing switches using adaptive body biasing technique," in *2009 52nd IEEE International Midwest Symposium on Circuits and Systems*, 2009, pp. 447–450.

[86] Z. Seifoori, H. Asadi, and M. Stojilović, "A machine learning approach for power gating the fpga routing network," in *2019 International Conference on Field-Programmable Technology (ICFPT)*, 2019, pp. 10–18.

[87] S. Chtourou, M. Abid, Z. Marrakchi, and H. Mehrez, "Power dissipation analysis for island-style fpga architecture," in *2014 5th International Conference on Information and Communication Systems (ICICS)*, 2014, pp. 1–4.

[88] A. A., A. H., E. T., H. J., and T. M., "Accurate thermal-profile estimation and validation for fpga-mapped circuits," in *Field-Programmable Custom Computing Machines (FCCM), 2013 IEEE 21st Annual International Symposium on*, 2013, pp. 57–60.

[89] Z. S., A. I., L. C., L. A., B. V., and T. O., "Robust self-calibrated dynamic voltage scaling in fpgas with thermal and ir-drop compensation," in *IEEE Transactions on Power Electronics*, vol. 33, 2018, p. 8500–8511.

[90] B. Khaleghi, S. Salamat, M. Imani, and T. Rosing, "Fpga energy efficiency by leveraging thermal margin," in *2019 IEEE 37th International Conference on Computer Design (ICCD)*, 2019, pp. 376–384.

[91] S. Chandrasekaran and A. Amira, "Power reduction for fpga implementations : Design optimisation and high level modelling," in *2006 International Conference on Field Programmable Logic and Applications*, 2006, pp. 1–2.

[92] S. Mittal, "A survey of techniques for approximate computing," *ACM Computing Surveys*, vol. 48, no. 4, pp. 1–33, May 2016.

[93] C. Jaegul and P. Haesun, "Customizing computational methods for visual analytics with big data," *IEEE Computer Graphics and Applications*, vol. 33, no. 4, pp. 22–28, 2018.

[94] H. Younes, A. Ibrahim, M. Rizk, and M. Valle, "Algorithmic level approximate computing for machine learning classifiers," in *2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, 2019, pp. 113–114.

[95] M. Samadi, J. Lee, D. A. Jamshidi, A. Hormati, and S. Mahlke, "Sage: Self-tuning approximation for graphics engines," in *Proceedings of the Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2013, pp. 13–24.

[96] A. Vladyslav, "2018 ieee second international conference on data stream mining & processing (dsmp)," in *One Approach of Approximation for Incoming Data Stream in IoT Based Monitoring System*, 2018.

[97] V. Chippa, D. Mohapatra, K. Roy, S. Chakradhar, and A. Raghunathan, "Scalable effort hardware design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 9, pp. 2004–2016, 2014.

[98] A. Sampson, W. Dietl, E. Fortuna, D. Gnanapragasam, L. Ceze, and D. Grossman, "Enerj: Approximate data types for safe and general low-power computation," in *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, 2011, p. 164–174. [Online]. Available: https://doi.org/10.1145/1993498.1993518

[99] S. Lee, L. K. John, and A. Gerstlauer, "High-level synthesis of approximate hardware under joint precision and voltage scaling," in *Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, 2017, pp. 187–192.

[100] C. Li, W. Luo, S. S. Sapatnekar, and J. Hu, "Joint precision optimization and high level synthesis for approximate computing," in *Proceedings of the ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2015.

[101] F. Vaverka, R. Hrbacek, and L. Sekanina, "Evolving component library for approximate high level synthesis," in *Proceedins of the IEEE Symposium Series on Computational Intelligence (SSCI)*, 2016, pp. 1–8.

[102] S. Sidiroglou-Douskos, S. Misailovic, H. Hoffmann, and M. Rinard, "Managing performance vs. accuracy trade-offs with loop perforation," in *Proceedings of the ACM SIGSOFT Symposium and the European Conference on Foundations of Software Engineering (ESEC/FSE)*, 2011, p. 124–134. [Online]. Available: https://doi.org/10.1145/2025113.2025133

[103] A. Rahimi, L. Benini, and R. Gupta, "Spatial memoization: Concurrent instruction reuse to correct timing errors in simd architectures," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 60, no. 12, pp. 847–851, 2013.

[104] G. Inigo, B. Ricardo, N. Santosh, and N. Thu, "Approxhadoop: Bringing approximations to mapreduce frameworks," *ACM SIGPLAN Notices*, vol. 50, no. 4, pp. 383–397, 2015.

[105] T. Mattia, P. Gianluca, and P. Christian, "Dynamically-tunable dataflow architectures based on markov queuing models," *Electronics*, p. 6, 2022.

[106] M. Sparsh, "A survey of architectural techniques for improving cache power efficiency," in *Sustainable Computing: Informatics and Systems*, vol. 4, 2014, pp. 33–43.

[107] S. Su, Y. Wu, and W. Qian, "Efficient batch statistical error estimation for iterative multi-level approximate logic synthesis," in *Proceedings of the ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2018. [Online]. Available: https://doi.org/10.1145/3195970.3196038

[108] K. Nepal, Y. Li, R. I. Bahar, and S. Reda, "Abacus: A technique for automated behavioral synthesis of approximate computing circuits," in *Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2014, pp. 1–6.

[109] O. Segal, M. Margala, S. R. Chalamalasetti, and M. Wright, "High level programming framework for fpgas in the data center," in *2014 24th International Conference on Field Programmable Logic and Applications (FPL)*, 2014, pp. 1–4.

[110] M. Gao and G. Qu, "Energy efficient runtime approximate computing on data flow graphs," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2017, pp. 444–449.

[111] D. Garg, K. Sharma, and A. Singla, "Designing a green data processing device using different input/output standards on fpga," in *2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, 2018, pp. 75–79.

[112] B. Christophe, M. Joel Mandebi, C. Paul, and at all, "The future of fpga acceleration in datacenters and the cloud." in *2ACM Transactions on Reconfigurable Technology and Systems*, vol. 15, 2022, pp. 1–42. [Online]. Available: https://doi.org/10.1145/3506713

[113] E. C. Andrew Putnam, Adrian Caulfield, "A reconfigurable fabric for accelerating large-scale data center services." in *IEEE Micro.*, 2015, pp. 10–22. [Online]. Available: https://doi.org/10.1109/MM.2015.42

[114] E. Dieter, "Competing in artificial intelligence chips: China's challenge amid technology war." in *Centre for International Governance Innovation, Special Report. 10/2020*, 2020, pp. 1–3.

[115] X. Yu, J. Gao, Y. Wang, J. Miao, E. Wu, H. Zhang, Y. Meng, B. Zhang, B. Min, and D. Chen, "A data-center FPGA acceleration platform for convolutional neural networks," in *29th International Conference on Field Programmable Logic and Applications, FPL 2019, Barcelona, Spain, September 8-12, 2019*, I. Sourdis, C. Bouganis, C. Álvarez, L. A. T. Díaz, P. Valero-Lara, and X. Martorell, Eds. IEEE, 2019, pp. 151–158. [Online]. Available: https://doi.org/10.1109/FPL.2019.00032

[116] A. Francois, W. Jagath, H. Christoph, W. Beat, and P. Stephan, "An fpga platform for hyperscalers." in *IEEE 25th Annual Symposium on High-Performance Interconnects.*, 2017, pp. 29–32.

[117] E. Chung, J. Fowers, K. Ovtcharov, M. Papamichael, A. Caulfield, T. Massengill, M. Liu, S. Alkalay, and M. Haselman, "Serving dnns in real time at datacenter scale with project brainwave." in *IEEE Micro*, vol. 38, 2018, pp. 8–20.

[118] G. A. D., H. M. C., L. H., L. A. G., S. J., and Y. C., "Novo-g#: Large-scale reconfigurable computing with direct and programmable interconnects." in *IEEE High Performance Extreme Computing Conference (HPEC).*, 2016, pp. 1–7.

[119] G. Alan, L. Herman, and S. Greg, "Novo-g: At the forefront of scalable reconfigurable supercomputing." in *Comput. Sci. Eng.*, vol. 13, 2010, pp. 82–86.

[120] C. Chris, T. Ian, E. D., A. Vikas, and G. A., "Narc: Network attached reconfigurable computing for high performance, network based applications." in *8th Annual International Conference on Military and Aerospace Programmable Logic Devices (MAPLD'05).*, 2005, pp. 1–10.

[121] B. R., B. S., B. M., C. G., P. J., P. M., S. A., T. A., M. A., s. G., S. R., C. A., C. R., and G. G., "Maxwell – a 64 fpga supercomputer." in *2nd NASA/ESA Conference on Adaptive Hardware and Systems (AHS'07).*, 2007, p. 287–294.

[122] L. Thomas, P. Byungchul, B. Hadi, and L.-G. Alberto, "Savi testbed architecture and federation. in future access enablers of ubiquitous and intelligent infrastructures." in *Springer.*, 2015, pp. 3–10.

[123] A. P. Adrian Caulfield, Eric Chung, "A cloud-scale acceleration architecture." in *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO).*, 2016. [Online]. Available: https://doi.org/10.1109/MICRO.2016.7783710

[124] J. Weerasinghe, R. Polig, F. Abel, and C. Hagleitner, "Network-attached fpgas for data center applications," in *2016 International Conference on Field-Programmable Technology (FPT)*, 2016, pp. 36–43.

[125] J. Hoozemans, J. Peltenburg, F. Nonnemacher, A. Hadnagy, Z. Al-Ars, and H. P. Hofstee, "Fpga acceleration for big data analytics: Challenges and opportunities," *IEEE Circuits and Systems Magazine*, vol. 21, no. 2, pp. 30–47, 2021.

[126] I. Stamelos, E. Koromilas, C. Kachris, and D. Soudris, "A novel framework for the seamless integration of fpga accelerators with big data analytics frameworks in heterogeneous data centers," in *2018 International Conference on High Performance Computing Simulation (HPCS)*, 2018, pp. 539–545.

[127] A. Saghir and T. Masood, "Performance evaluation of openstack networking technologies," in *2019 International Conference on Engineering and Emerging Technologies (ICEET)*, 2019, pp. 1–6.

[128] A. Pereira and R. Sinnott, "A performance evaluation of containers running on managed kubernetes services," in *2019 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, 2019, pp. 199–208.

[129] C. Bobda, A. Majer, A. Ahmadinia, T. Haller, A. Linarth, and J. Teich, "The erlangen slot machine: increasing flexibility in fpga-based reconfigurable platforms," in *Proceedings. 2005 IEEE International Conference on Field-Programmable Technology, 2005.*, 2005, pp. 37–42.

[130] N. Gil, S. Amy, L. Felix, R. Dion, and U. Rich, "Intel virtualization technology: Hardware support for efficient processor virtualization," in *Intel Technol. J.*, 2006, p. 10.

[131] J. Guo, L. Zhang, J. R. Hung, C. Li, J. Zhao, and M. Guo, "FPGA sharing in the cloud: a comprehensive analysis," *Frontiers Comput. Sci.*, vol. 17, no. 5, p. 175106, 2023. [Online]. Available: https://doi.org/10.1007/s11704-022-2127-0

[132] S. Hayden, Kwok-Hay and L. Cheng, "Fpgas for software programmers," in *Springer*, 2016, pp. 285–305.

[133] I. Gonzalez, S. Lopez-Buedo, G. Sutter, D. Sanchez-Roman, F. J. Gomez-Arribas, and J. Aracil, "Virtualization of reconfigurable coprocessors in hprc systems with multicore architecture," *Journal of Systems Architecture*, vol. 58, no. 6, pp. 247–256, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383762112000197

[134] A. A. Al-Aghbari and M. E. S. Elrabaa, "Cloud-based fpga custom computing machines for streaming applications," *IEEE Access*, vol. 7, pp. 38 009–38 019, 2019.

[135] G. Dai, Y. Shan, F. Chen, Y. Wang, K. Wang, and H. Yang, "Online scheduling for fpga computation in the cloud," in *2014 International Conference on Field-Programmable Technology (FPT)*, 2014, pp. 330–333.

[136] Z. Ke, C. Yisong, C. Mingyu, B. Yungang, and X. Zhiwei, "Computer organization and design course with fpga cloud," in *50th ACM Technical Symposium on Computer Science Education. ACM*, 2019, p. 927–933.

[137] A. Brant and G. G. Lemieux, "Zuma: An open fpga overlay architecture," in *2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines*, 2012, pp. 93–96.

[138] S. Ma, Z. Aklah, and D. Andrews, "A run time interpretation approach for creating custom accelerators," in *2015 25th International Conference on Field Programmable Logic and Applications (FPL)*, 2015, pp. 1–4.

[139] S. Moini, S. Tian, D. Holcomb, J. Szefer, and R. Tessier, "Power side-channel attacks on bnn accelerators in remote fpgas," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 11, no. 2, pp. 357–370, 2021.

[140] A. Boutros, M. Hall, N. Papernot, and V. Betz, "Neighbors from hell: Voltage attacks against deep learning accelerators on multi-tenant fpgas," in *International Conference on Field-Programmable Technology, (IC)FPT 2020, Maui, HI, USA, December 9-11, 2020*. IEEE, 2020, pp. 103–111. [Online]. Available: https://doi.org/10.1109/ICFPT51103.2020.00023

[141] F. Schellenberg, D. R. Gnad, A. Moradi, and M. B. Tahoori, "An inside job: Remote power analysis attacks on fpgas," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2018, pp. 1111–1116.

[142] Z. Weissman, T. Tiemann, D. Moghimi, E. Custodio, T. Eisenbarth, and B. Sunar, "Jackhammer: Efficient rowhammer on heterogeneous fpga-cpu platforms," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, no. 3, p. 169–195, Jun. 2020. [Online]. Available: https://tches.iacr.org/index.php/TCHES/article/view/8587

[143] M. K. Ahmed, J. Mandebi, S. K. Saha, and C. Bobda, "Multi-tenant cloud FPGA: A survey on security," *CoRR*, vol. abs/2209.11158, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2209.11158

[144] E. Karabulut, C. Yuvarajappa, M. I. Shaik, S. Potluri, A. Awad, and A. Aysu, "Pr crisis: Analyzing and fixing partial reconfiguration in multi-tenant cloud fpgas," in *Proceedings of the 2022 Workshop on Attacks and Solutions in Hardware Security*, ser. ASHES'22. New York, NY, USA: Association for Computing Machinery, 2022, p. 101–106. [Online]. Available: https://doi.org/10.1145/3560834.3563832

[145] Z. Y., W. Y., and W. X., "Capping the electricity cost of cloud-scale data centers with impacts on power markets," in *Proceedings of the 20th international symposium on High performance distributed computing*, 2011, pp. 271–272.

[146] A. Yarahmadi and M. Momtazpour, "Vm placement in accelerator-equipped data centers using variable-length modified genetic algorithm," in *2021 29th Iranian Conference on Electrical Engineering (ICEE)*, 2021, pp. 562–567.

[147] S. Zhang, F. Meng, and Z. Zhang, "A cloud data center virtual machine placement scheme based on energy optimization," in *2018 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, 2018, pp. 215–2156.

[148] M. Bertolino, R. Pacalet, L. Apvrille, and A. Enrici, "Efficient scheduling of fpgas for cloud data center infrastructures," in *2020 23rd Euromicro Conference on Digital System Design (DSD)*, 2020, pp. 57–64.

[149] N. Gebara, J. Meng, W. Luk, and P. Costa, "Scheduling algorithms for high performance network switching on fpgas: A survey," in *2018 International Conference on Field-Programmable Technology (FPT)*, 2018, pp. 166–173.

[150] S. P and V. P, "A green energy optimized scheduling algorithm for cloud data centers," in *International Conference on Computing and Network Communications (CoCoNet)*, 2015, pp. 941–945.

[151] A. Dhar, E. Richter, M. Yu, W. Zuo, X. Wang, N. S. Kim, and D. Chen, "Dml: Dynamic partial reconfiguration with scalable task scheduling for multi-applications on fpgas," *IEEE Transactions on Computers*, pp. 1–1, 2021.

[152] W. T. Loke and C. Y. Koay, "Energy-aware scheduling for task adaptive fpgas," in *2016 International Conference on Field-Programmable Technology (FPT)*, 2016, pp. 173–176.

[153] R. Huigui, Z. Haomin, X. Sheng, L. Canbing, and H. Chunhua, "Optimizing energy consumption for data centers," in *Renewable and Sustainable Energy Reviews*, vol. 58, 2016, pp. 674–691.

[154] B. d. A. Silva and V. Bonato, "Power/performance optimization in fpga-based asymmetric multi-core systems," in *22nd International Conference on Field Programmable Logic and Applications (FPL)*, 2012, pp. 473–474.

[155] P. Grünwald and P. M. B. Vitányi, "Shannon information and kolmogorov complexity," *CoRR*, vol. cs.IT/0410002, 2004. [Online]. Available: http://arxiv.org/abs/cs.IT/0410002

[156] A. Radovanovic, R. Koningstein, I. Schneider, B. Chen, A. Duarte, B. Roy, D. Xiao, M. Haridasan, P. Hung, N. Care, S. Talukdar, E. Mullen, K. Smith, M. Cottman, and W. Cirne, "Carbon-aware computing for datacenters," *IEEE Transactions on Power Systems*, pp. 1–1, 2022.

[157] Microsoft. (2020) Carbon-aware kubernetes. [Online]. Available: https://devblogs.microsoft.com/sustainable-software/carbon-aware-kubernetes/

[158] Google. (2021) New carbon-intelligent computing platform. [Online]. Available: https://blog.google/inside-google/infrastructure/data-centers-work-harder-sun-shines-wind-blows/

Mattia Tibaldi is a Ph.D. candidate at Politecnico di Milano. He received a Master's degree in Computer Science and Engineering from the same university in 2021. His research interests include reconfigurable and adaptive systems for cloud applications and data centers, security-aware design methodologies, and sustainable computing.

Christian Pilato is an Associate Professor at Politecnico di Milano. He was a Post-doc Research Scientist at Columbia University (2013-2016) and Università della Svizzera italiana (2016-2018), and an Assistant Professor at Politecnico di Milano (2018-2023). He was also a Visiting Researcher at New York University, TU Delft, and Chalmers University of Technology. He has a Ph.D. in Information Technology from Politecnico di Milano (2011). His research interests include high-level synthesis, reconfigurable systems, and system-on-chip architectures, focusing on memory and security aspects. He served as program chair of EUC 2014 and ICCD 2022 and is currently serving on the program committees of many conferences on EDA, CAD, embedded systems, and reconfigurable architectures (like DAC, ICCAD, DATE, CASES, FPL, ICCD, etc.) He is a Senior Member of IEEE and ACM, and a Member of HiPEAC.