Power reduction for the RISC-V core version CV32A6 implemented on FPGA

Matthieu Laidet, Isaure Badier, Guillaume De Freitas

***Abstract*—The RISC-V open-source Instruction Set Architecture (ISA) is increasingly gaining in popularity. Industry actors adapt RISC-V processor cores to best fit their needs. When flexibility is key — or for small batches — an implementation on Field Programmable Gate Array (FPGA) is well suited. To this end, Thales Research France derived from the RISC-V softcore CVA6 version a new softcore : CV32A6. To further reduce its energy consumption while running Machine learning tasks, they organized a student contest to gather ideas and solutions. This paper explores energy consumption reduction of the RISC-V core CV32A6 version implemented on FPGA, as part of the *2nd national RISC-V student contest: Power optimization of the CV32A6 RISC-V softcore*. Our work focuses on hardware level optimization, not on ISA optimization.**

First we present the CV32A6 softcore, the Convolutionnal Neural Network (CNN) based application run on the core and applicable techniques to reduce energy consumption. Then we go on exploring said techniques on a Zybo Z7 ZYNQ-7020 development board, including Clock-gating, Datapath blocking, Region-Constrained Placement and Power-gating. We did not manage to reduce energy consumption within the scope of the contest. Our experimentation indicate that the hardware architecture has already been thoroughly optimized.

***Index Terms*—FPGA, Power Reduction, Hardware level optimization, CVA6 (Ariane), Zynq System On Chips, Clock-gating, Power-gating, Region Constrained Placement.**

CONTENTS

I. INTRODUCTION

A. The RISC-V core, CVA6 version

RISC-V architecture is an ISA provided under open sources licenses, initially developed to support computer architecture research and education [1]. Its popularity keeps increasing as the free and open-source character allows modification and extensions of existing implementations. Therefore RISC-V is well-suited for niche application such as the space domain where reliability is key [2]. In 2018, ETH Zürich improved the design by adding a memory management unit and the possibility to support an Linux Operating System in a new RISC-V version named *Ariane* [3]. A year later, the company OpenHW Group created the CVA6 version, which includes and replaces Ariane.

The RISC-V core, CVA6 version, is a 6-stage, single issue, in-order Central Processing Unit (CPU) implementing the RV64I instruction set including the M, A and C extensions [4][1]. The CVA6 version was primarily designed for Application Specific Integrated Circuits (ASIC) targets. Still, the CVA6 version is a popular application-class processor when it comes to FPGA implementation . This type of implementation provides a flexible solution for prototyping and allows developers to update their designs. Thales Research and Technology France, in collaboration with the CNFM and the GDR SOC^2 created a contest to improve power consumption of CV32A6 which is a 32 bit variant of CVA6 created by Thales [3].

B. The development board

The board used for the testing in the contest is the Zybo Z7 ZYNQ-7020 (Z7-20) development board. It features a XC7Z020-1CLG400C Xilinx System On Chip (SOC) [5] of which the programmable logic is equivalent to an Artix-7 FPGA [6]. Hence the focus of this state of the art on techniques applicable to Xilinx FPGAs.

C. The application run on the softcore

In the contest, the same application is used to measure power throughout the entire contest. It can be valuable to understand its behavior to further improve energy consumption.

1) The MNIST Database: The MNIST database is a set of more than 70,000 handwritten digits. It is a subset of the NIST database in which the pictures have been normalized with regards to centering and size [7]. Handwritten digit recognition is often used as a test case for pattern recognition and machine learning algorithms [8].

2) The application: A simple CNN aiming at recognizing the handwritten numbers of the MNIST database is run on the softcore. Particularly, in the current version of the application, only *env0003* (a picture of a *4*) is tested.
The CNN has four layers, two convolutionnal layers and two fully connected layers (figure 1) [9]. The use of a CNN relies on multiplication based operations run in loop [10].

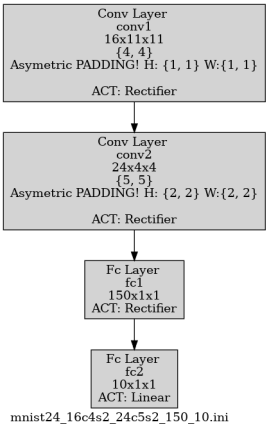


Fig. 1: Visual representation of the MNIST application used for power measurement [9]

D. Power and energy definitions

The power consumption of an electronic circuit at the logic level is classified in two distinct parts. The dynamic power is caused by the switching of the transistors, and the static power is due to the biasing of the transistors [11]. For a long time, dynamic power was mainly responsible for the power consumption but with the reduction in semiconductor process size, the dynamic power's order of magnitude decreased such as static power is no longer negligible.

1) Dynamic power:

$$P_{dyn} = a \cdot C \cdot V^2 \cdot f \tag{1}$$

Dynamic power is defined in equation 1, where a is the activity factor (the proportion of transistors switching per unit of time), C is the load capacitance, V the supply voltage and f the frequency of the clock. This equation shows that lowering the frequency has a linear impact on dynamic power. Due to the quadratic relation between supply voltage and power, voltage reduction is more effective.

2) Static power:

$$P_{static} = V \cdot I_{leak} \tag{2}$$

Static power is defined by equation 2, where I_{leak} is the leakage current and V the voltage. To operate, transistors need to be biased. As long as there is a drain-to-source voltage, the body diode of the transistor is reverse-biased which causes leakage current. In CMOS technology, this leakage current can be approximated by equation 3.

$$I_{leak} \approx K_1 W e^{-V_{th}/nV_\theta}(1 - e^{-V/V_\theta}) \tag{3}$$

with K_1, V_θ, n constants at a given temperature, W the gate width, V the command voltage and V_{th} threshold voltage of the transistor (lower than V). To reduce the leakage current, one can only shut off the voltage V since increasing V_{th} is impossible as it is fixed by the CMOS technology.

E. Static power reduction techniques

1) Region-Constrained Placement: On Xilinx FPGAs, the internal structure can be divided in regions composed of one, or several Configurable Logic Blocks (CLBs) sharing the same power switch. Region-Constrained Placement (RCP) aims at reducing the number of occupied regions thus reducing the number of powered but barely used regions (figure 2). The fewer powered regions there is, the smaller the leakage current is. The improvements on static leakage power can be up to 13% making this solution worth while to explore [12]. Regarding the implementation, a User Constraints File (UCF) can limit the placement to a specific number of regions.

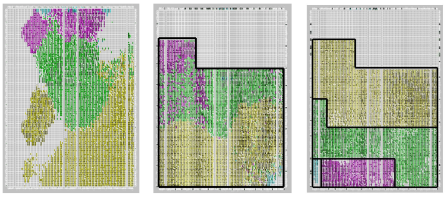


Fig. 2: Different placements for an example design [12]

2) Power-gating: To decrease leakage current and static power, it is possible to cut power to a logic block by adding a sleep transistor between the block and the supply rail (figure 3).

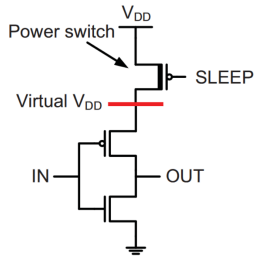
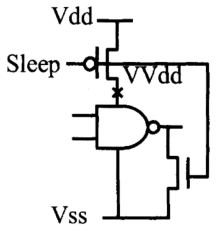


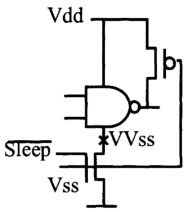
Fig. 3: Example of power-gating [13]

If the transistor is between V_{DD} and the logic block, it is a PMOS header switch (figure 4a) whereas it is a NMOS footer

switch if it is between the block and ground (figure 4b) [14]. A $Sleep$ signal (or $\overline{Sleep}$) is used to enable or disable power on the logical block.



(a) Header sleep transistor



(b) Footer sleep transistor

Fig. 4: Types of sleep transistor for power-gating [14]

Power-gating is an efficient way to reduce static power but shows a major disadvantage. First, it must be part of the FPGA architecture. If so, when the circuit is biased again, the inrush current causes supply voltage drops and disturbs the behavior of the circuit [13]. The solution is to add decoupling capacitors, which increases the size of the circuit [15]. Power-gating circuitry is available in Zynq SOCs. Hosseinabady and Nunez-Yanez developed a technique for run-time power-gating which relies on the Cortex A9 processor to control the circuitry dynamically [16].

F. Dynamic power reduction techniques

1) Clock-Gating: Clock-gating is a method for decreasing dynamic power consumption. It consists in turning off clock signals for unused parts of a system. Hence it stops unnecessary switching activity [17]. To do so, the enable signal is applied both to the clock signal and the actual component (figure 5).

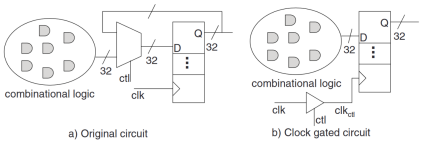
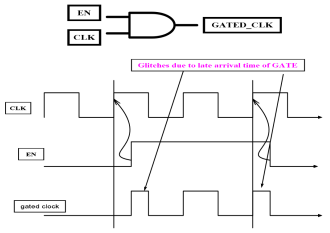
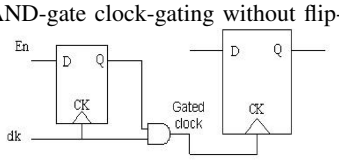


Fig. 5: Clock-gating example for FPGA structure [18]

The simplest way is to use an AND-gate between the clock signal and the enable signal and to feed its output to the circuit. However, to avoid glitches, a flip-flop needs to be connected to the AND-gate. Indeed, when the enable signal goes high in the middle of the clock cycle, the gated clock glitches (see in the timing diagram in figure 6a). To avoid this, a D flip-flop is added before the AND-gate (figure 6b).
Clock-gating always improves the power profile and shows better results as the number of flip-flops in the design increases [19].



(a) AND-gate clock-gating without flip-flop



(b) AND-gate clock-gating with flip-flop

Fig. 6: Efficient clock-gating to avoid glitches [20]

2) Blocking the Datapath: To avoid dynamic power consumption for an enabled but unused part of the system, one can also block its datapath thus preventing any flip-flop to switch state. However this technique is harder to implement and is not as efficient as others such as clock-gating when a large number of flip-flop is used [19].

3) Idle-width switching activity: A sub case of datapath blocking allows the use of narrow bitwidth operations. Applications treating sound and images often only require 16 bits operations [21].

The upper bits' inputs of a component are latched. At run time, if the upper bits are not needed, the latch is disabled and this portion of input data is not updated. This reduces dynamic power as less updating on the input results in less switching activity. Figure 7 is a proposition of datapath implementation for an ALU using this method for the 48 upper bits. In the case of narrow bitwidth operations, the 48 upper bits of the result are set to 0 [21].

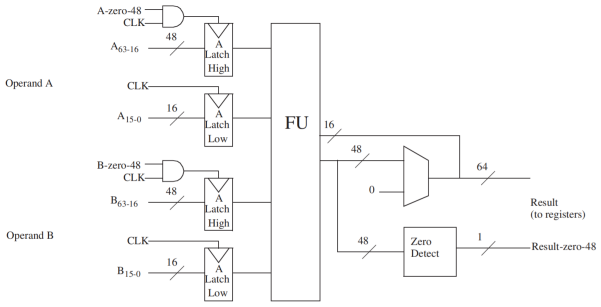


Fig. 7: ALU for narrow width operand [22]

Another aspect is to detect narrow bitwidth operation. It can be done at the instruction level if a specific instruction requires

only lower bits. It can also be done directly on the operand with a zero detection logic. [21]

G. Discussion

The most efficient power reduction techniques presented in this paper operate outside of the programmable logic level of the FPGA. They are suitable for FPGA powered devices. The objective of the contest being to provide power optimization to the CV32A6 softcore, we have to restrain our work to System-Level methods (which can be described in SystemVerilog). Techniques such as run-time power-gating cannot be implemented because it relies either on the Cortex A9 featured in the SoC, or it requires external circuitry. Clock-gating, Region-Constrained Placement, idle-width switching activity and static power-gating are the only applicable techniques within the scope of the contest.

II. Methodology

A. Objectives of the contest

The goal of the contest is to optimize the total energy (equation 4) consumption for the aforementioned application.

$$E = P_{dyn} \cdot t_{dyn} + P_{static} \cdot t_{static} \tag{4}$$

The simulation tools used for the power analysis give us the total power i.e. the sum of static and dynamic power [9]. Hence, the total energy is given by equation 5, with N the number of clock cycles and T the clock period.

$$E = P_{tot} \cdot t = P_{tot} \cdot N \cdot T \tag{5}$$

B. Our approach

Our field of study is automation and electronics, with a specialization in embedded systems. Our background is more oriented towards electrical engineering rather than computer science. Therefore, we focus on hardware optimization.

When simulating the reference design with Questa, we notice three different phases. First, the application receives the binary file of the application. Then comes the computation and finally the transmission of the results over the UART. During the computation, the multiplier unit is used extensively for the convolution [10]. However, it is inactive during the first and last part of the application's execution.

According to this observation, we decide to implement clock-gating to reduce the switching activity of the multiplication unit when not needed. The objective is to create a clock-gating specific sub-module. This sub-module should be part of the ex-stage and should handle this technique without modifying the global architecture.

Out of the contest's scope, Region Constraint Placement in combination with static power-gating is studied as a means of reducing static power. As the design does not use all the resources of the FPGA, we forbid placement in a specific clock region. By shutting down power to this area using Xilinx constraints, power is saved as the clock tree is not driven anymore in this area.

III. Results

A. Post routing power analysis and timing report for the reference design

The reference values (fig. 9) are run at the default clock period ($T = 22.2ns$). From these we can derive the total energy (equation 6). These values are used to compute the variation over the reference design (Δ*/ref.*) in the other implementation's results.

$$E_{tot} = P_{tot} \cdot N \cdot T = 0.307 \times 2098749 \times 22.2 \cdot 10^{-9} = 14.30mJ \tag{6}$$

Impl. Type	Measurement	Value (ns)
Real	Clock Period	22.2
	Worst Negative slack	0.538
	Worst Hold slack	0.023
Out of Context	Clock Period	22.2
	Worst Negative slack	3.024
	Worst Hold slack	0.205

Fig. 8: Timing report of the reference design

Measurement	Value
Total On-Chip Power (W)	0.307
Instructions	1725056
Clock cycles	2098749
LUT as Logic	26755
Register	22768
F7/F8 Muxes	1336
CARRY4	1323
LUT as Distributed RAM	52
LUT as Shift Register	121
Others	715
Signals	46725
Block RAM	86
MMCM	1
DSPs	4
I/O	9

Fig. 9: Power, performance and Cell usage of the reference design

B. Clock-gating module

1) Clock-gating implementation: Our clock-gating module is connected to the rest of the design as described in figure 11. Its input *sbe_clock_gating_i* is analyzed to determine whether to clock the multiplier or not. If the multiplier is required by the current instruction, the multiplier is clocked for one whole duration of the pipeline to ensure that the need for the multiplier is smoothed. Hence it avoids useless switching of the gating signal.

```
logic              needs_mult      =1'b0;
logic[2:0]         mult_counter    =0;

always_comb begin
needs_mult=(sbe_clock_gating_i.op >= MUL
        && sbe_clock_gating_i.op <= MULW)
                ? 1'b1:1'b0;
clk_mult=(mult_counter !=0 ) ? clk_i:1'b0;
end

always @ (posedge clk_i) begin
 if (needs_mult == 1) begin
        mult_counter = 1;
 end else if( mult_counter !=0) begin
        mult_counter++;
 end else begin
        mult_counter =0;
 end
end
```

Fig. 10: SystemVerilog code of the clock-gating unit

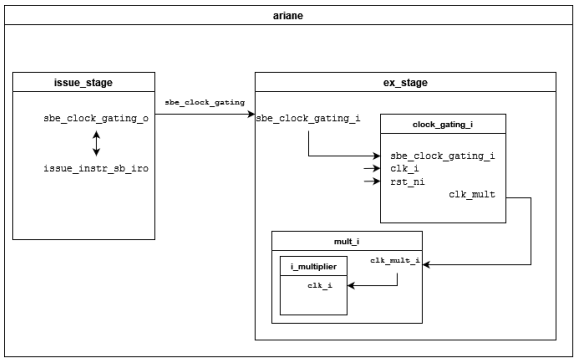


Fig. 11: Clock-gating module interconnections with all affected modules

2) Post routing power analysis and timing report: The clock-gating unit does not improve the total on-chip power and shows small changes to the cell usage.

Impl. Type	Measurement	Value (ns)	Δ/ref.
Real	Clock Period	22.2	-
	Worst Negative slack	0.563	+4.65%
	Worst Hold slack	0.016	-30%
Out of Context	Clock Period	22.2	-
	Worst Negative slack	2.531	-16.3%
	Worst Hold slack	0.119	-72.3%

Fig. 12: Timing report of the clock-gating unit featured design

Measurement	Value	Δ/ref.
Total On-Chip Power (W)	0.307	-
Instructions	1725056	-
Clock cycles	2098749	-
LUT as Logic	26724	-0.10%
Register	22769	$\approx 0\%$
F7/F8 Muxes	1321	-1.12%
CARRY4	1323	-
LUT as Distributed RAM	52	-
LUT as Shift Register	121	-
Others	717	$+2.8\%$
Signals	46592	-0.30%
Block RAM	86	-
MMCM	1	-
DSPs	4	-
I/O	9	-

Fig. 13: Power, performance and Cell usage of the clock-gating module featured design

C. Region-Constrained Placement

By adding the following constraint in `fpga/zyo_z7_20.xdc`, we were able to prohibit placement to an entire clock region.

```
set_property PROHIBIT TRUE
 [get_sites -of [get_clock_region X0Y2]]
```

Measurement	Value	Δ/ref.
Total On-Chip Power (W)	0.306	-0.33%
Instructions	1725056	-
Clock cycles	2098749	-
LUT as Logic	26750	-0.01%
Register	22768	-
F7/F8 Muxes	1336	-
CARRY4	1323	-
LUT as Distributed RAM	52	-
LUT as Shift Register	121	-
Others	715	-
Signals	46730	$+0.01\%$
Block RAM	86	-
MMCM	1	-
DSPs	4	-
I/O	9	-
Clock Period	22.2	-
Worst Negative slack	0.715	+32.9%
Worst Hold slack	0.021	-8.70%

Fig. 14: Power, performance, Cell usage and Timing Report of the RCP implementation

IV. Discussion

As showed in table 13, the clock-gating unit does not improve the overall power. Hence, for the same clock period, the total energy for a frame is the same as the reference design. However, timing-wise we are unable to analyse the results. By routing the clock through combinatory module we expect a longer critical path. That is shown by the out-of-context implementation's timing report but not by the real timing report that takes I/O constraints into account (figure 12). As measure of transparency we would not consider the reduction of the WNS in the clock-gating module featured implementation as reliable.

Forbidding placement to an entire clock region saves 1 mW on the total on-chip power (fig. 14). Due to our uncertainty on the timing reports we are unable to provide an analysis on the impact of the placement on timings. Common-sense suggest us that it would reduce slack since we don't allow the placer to work in its most efficient way. Nevertheless, this was not our objective, we did not manage to combine region constrained placement with static power-gating. Although our research showed that it was possible years ago on Xilinx Virtex-II [12] and Spartan3 FPGAs [23], this often requires a revisited FPGA architecture or external circuits to implement power-gating, which is out of the contest's scope. Regarding the ZYNQ-7020 SoC featured on our development board, a software-controlled power-gating may be achieve with the ARM Cortex-A9 embedded in the on-chip architecture [16].

The hardware architecture of the RISC-V core, CV32A6 version is already thoroughly optimized. More complex techniques may be implemented to save power, such as run-time power-gating or Dynamic Voltage Scaling [24], but they are far beyond the contest's scope. Therefore, our focus on hardware optimization lead us on a wrong track regarding the contest. Redesigning the instruction set architecture could have been more profitable energy-wise.

References

[1] A. Waterman, K. Asanovic, and C. Division, "The RISC-V Instruction Set Manual. Volume I: User-Level ISA," en, p. 145, May 2017. [Online]. Available: https://riscv.org/wp-content/uploads/2017/05/riscv-spec-v2.2.pdf (visited on 02/17/2022).

[2] A. Dörflinger, M. Albers, B. Kleinbeck, *et al.*, "A comparative survey of open-source application-class RISC-V processor implementations," in *Proceedings of the 18th ACM International Conference on Computing Frontiers*, ser. CF '21, New York, NY, USA: Association for Computing Machinery, May 2021, pp. 12–20, ISBN: 978-1-4503-8404-9. DOI: 10.1145/3457388.3458657. [Online]. Available: https://doi.org/10.1145/3457388.3458657 (visited on 12/03/2021).

[3] *CVA6 softcore contest*, original-date: 2020-10-14T13:50:28Z, Nov. 2021. [Online]. Available: https://github.com/ThalesGroup/cva6-softcore-contest (visited on 01/18/2022).

[4] F. Zaruba and L. Benini, "The Cost of Application-Class Processing: Energy and Performance Analysis of a Linux-ready 1.7GHz 64bit RISC-V Core in 22nm FDSOI Technology," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 11, pp. 2629–2640, Nov. 2019, arXiv: 1904.05442, ISSN: 1063-8210, 1557-9999. DOI: 10.1109/TVLSI.2019.2926114. [Online]. Available: http://arxiv.org/abs/1904.05442 (visited on 12/03/2021).

[5] Digilent, *Zybo Z7 Reference Manual - Digilent Reference*. [Online]. Available: https://digilent.com/reference/programmable-logic/zybo-z7/reference-manual (visited on 01/26/2022).

[6] XILINX, *Zynq-7000 SoC Data Sheet: Overview (DS190)*, en, 2018. [Online]. Available: https://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf.

[7] Y. LeCun, C. Cortes, and C. J. Burges, *MNIST handwritten digit database*, en. [Online]. Available: http://yann.lecun.com/exdb/mnist/ (visited on 02/01/2022).

[8] L. Deng, "The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, Nov. 2012, Conference Name: IEEE Signal Processing Magazine, ISSN: 1558-0792. DOI: 10.1109/MSP.2012.2211477. [Online]. Available: https://ieeexplore.ieee.org/document/6296535.

[9] Thales, *2nd national RISC-V student contest 2021-2022 : Reporting energy results*, en, original-date: 2020-10-14T13:50:28Z, Nov. 2021. [Online]. Available: https://github.com/ThalesGroup/cva6-softcore-contest/blob/b80d2d111d9d5342a50ea5115062e67e30dbb491/Reporting_energy_results_v2.pdf (visited on 01/24/2022).

[10] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Tech-*

nology (ICET), Aug. 2017, pp. 1–6. DOI: 10.1109/ICEngTechnol.2017.8308186.

[11] N. Kim, T. Austin, D. Baauw, *et al.*, "Leakage current: Moore's law meets static power," *Computer*, vol. 36, no. 12, pp. 68–75, Dec. 2003, Conference Name: Computer, ISSN: 1558-0814. DOI: 10.1109/MC.2003.1250885. [Online]. Available: https://ieeexplore.ieee.org/document/1250885.

[12] A. Gayasen, Y. Tsai, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, and T. Tuan, "Reducing leakage energy in FPGAs using region-constrained placement," in *Proceedings of the 2004 ACM/SIGDA 12th international symposium on Field programmable gate arrays*, ser. FPGA '04, New York, NY, USA: Association for Computing Machinery, Feb. 2004, pp. 51–58, ISBN: 978-1-58113-829-0. DOI: 10.1145/968280.968289. [Online]. Available: https://doi.org/10.1145/968280.968289 (visited on 01/20/2022).

[13] 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*, Dec. 2010, pp. 1–8. DOI: 10.1109/FPT.2010.5681533. [Online]. Available: https://ieeexplore.ieee.org/document/5681533.

[14] K. Shi and D. Howard, "Challenges in sleep transistor design and implementation in low-power designs," in *2006 43rd ACM/IEEE Design Automation Conference*, ISSN: 0738-100X, Jul. 2006, pp. 113–116. DOI: 10.1109/DAC.2006.229187. [Online]. Available: https://ieeexplore.ieee.org/document/1688772.

[15] H. Jiang, M. Marek-Sadowska, and S. Nassif, "Benefits and costs of power-gating technique," in *2005 International Conference on Computer Design*, ISSN: 1063-6404, Oct. 2005, pp. 559–566. DOI: 10.1109/ICCD.2005.34. [Online]. Available: https://ieeexplore.ieee.org/document/1524207.

[16] M. Hosseinabady and J. L. Nunez-Yanez, "Run-time power gating in hybrid ARM-FPGA devices," in *2014 24th International Conference on Field Programmable Logic and Applications (FPL)*, ISSN: 1946-1488, Sep. 2014, pp. 1–6. DOI: 10.1109/FPL.2014.6927503. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6927503.

[17] Y. Zhang, J. Roivainen, and A. Mammela, "Clock-Gating in FPGAs: A Novel and Comparative Evaluation," in *9th EUROMICRO Conference on Digital System Design (DSD'06)*, Aug. 2006, '584–590. DOI: 10.1109/DSD.2006.32. [Online]. Available: https://ieeexplore.ieee.org/document/1690091.

[18] 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*, ISSN: 1946-1488, Aug. 2009, pp. 112–118. DOI: 10.1109/FPL.2009.5272538. [Online]. Available: https://ieeexplore.ieee.org/document/5272538.

[19] J. P. Oliver, J. Curto, D. Bouvier, M. Ramos, and E. Boemo, "Clock gating and clock enable for FPGA power reduction," in *2012 VIII Southern Conference on Programmable Logic*, Mar. 2012, pp. 1–5. DOI: 10.1109/SPL.2012.6211782. [Online]. Available: https://ieeexplore.ieee.org/document/6211782.

[20] J. Shinde and S. S. Salankar, "Clock gating — A power optimizing technique for VLSI circuits," in *2011 Annual IEEE India Conference*, ISSN: 2325-9418, Dec. 2011, pp. 1–4. DOI: 10.1109/INDCON.2011.6139440. [Online]. Available: https://ieeexplore.ieee.org/document/6139440.

[21] D. Brooks and M. Martonosi, "Dynamically exploiting narrow width operands to improve processor power and performance," in *Proceedings Fifth International Symposium on High-Performance Computer Architecture*, Jan. 1999, pp. 13–22. DOI: 10.1109/HPCA.1999.744314. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=744314&tag=1.

[22] S. Kaxiras and M. Martonosi, "Computer Architecture Techniques for Power-Efficiency," *Synthesis Lectures on Computer Architecture*, vol. 3, no. 1, pp. 1–207, Jan. 2008, Publisher: Morgan & Claypool Publishers, ISSN: 1935-3235. DOI: 10.2200/S00119ED1V01Y200805CAC004. [Online]. Available: https://www.morganclaypool.com/doi/abs/10.2200/S00119ED1V01Y200805CAC004 (visited on 02/01/2022).

[23] R. Bharadwaj, R. Konar, P. Balsara, and D. Bhatia, "Exploiting temporal idleness to reduce leakage power in programmable architectures," in *Proceedings of the ASP-DAC 2005. Asia and South Pacific Design Automation Conference, 2005.*, ISSN: 2153-697X, vol. 1, Jan. 2005, 651–656 Vol. 1. DOI: 10.1109/ASPDAC.2005.1466244. [Online]. Available: https://ieeexplore.ieee.org/document/1466244.

[24] 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.*, Dec. 2005, pp. 173–180. DOI: 10.1109/FPT.2005.1568543. [Online]. Available: https://ieeexplore.ieee.org/document/1568543.