This article has been accepted for publication in IEEE Embedded Systems Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LES.2022.3212776

LoFFT: Low-voltage FFT Using Lightweight Fault Detection for Energy Efficiency

Mehdi Safarpour, *Member, IEEE,* Olli Silvén, *Senior Member, IEEE*

***Abstract*—Operating at reduced voltage is an effective technique for improving the energy efficiency of computing. However, the approach is constrained by its exacerbated sensitivity to Process, Voltage, and Temperature (PVT) variations, which under throughput constraints challenges finding the energy minimizing voltage-frequency operating point. Commonly utilized design approaches for adaptive voltage scaling are based on timing slack measurement or speculation techniques that require adding extra hardware, e.g., Error Detection Sequence (EDS) circuits, that substantially increase the design complexity, and are not applicable for already fabricated designs.**

In this paper, instead of circuit-level techniques, a low-cost algorithmic error detection method is proposed as the enabler for reduced voltage operation of Fast Fourier Transform (FFT) accelerators. Without requiring neither gate-level nor circuit-level modifications, the method works based on an intrinsic property of the Fourier transform, i.e., Parseval's identity. The method is demonstrated on a System-on-Chip (SoC) that integrates a Field-Programmable Gate Array (FPGA) made to operate at reduced voltages. The fault detection capability is profiled using the demonstration test bench, implemented both as software and as hardware. In the experiments, a ≈43% reduction in power consumption was achieved without sacrificing the throughput and reliability. The overheads of the proposed fault detection approach scale sub-linearly with respect to FFT size and are ≤10% for 1024-point FFT.

***Index Terms*—Low power, Near threshold, FFT processor.**

I. Introduction

ENERGY efficiency optimization is driving the trend toward accelerator-rich System-on-Chip architectures. In such schemes, dedicated hardware accelerators are employed to execute the most costly algorithms, including various transforms, neural network inference, matrix multiplication and decomposition [1], [2].

Fast Fourier Transform (FFT) is a fundamental operation in many signal processing applications ranging from wireless telecommunications to image processing [3], [2]. For example, Orthogonal Frequency-Division Multiplexing (OFDM) schemes in telecommunication systems rely on FFT for modulation and demodulation [2]. Similarly, FFTs are employed in accelerating large convolutional operations in Deep Neural Network (DNN) models [1]. This popularity, performance, and energy efficiency motivates designers to include FFT hardware accelerators in SoCs [3].

Since the introduction of the FFT algorithm in 1965 [5] numerous FFT methods have been proposed with focus on software and hardware implementations [3]. In this letter, the aim is to improve the energy efficiency of FFT hardware through operating at reduced voltages. The proposed approach does not necessitate circuit-level modifications, making it usable in existing accelerator-rich SoCs with multiple controllable voltage domains.

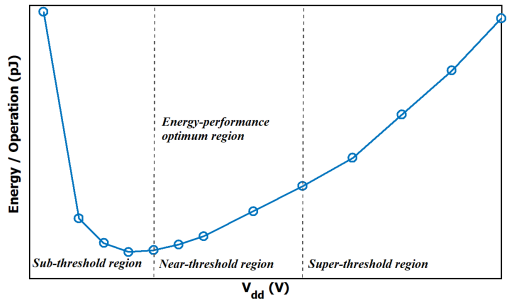

Fig. 1. Operating at reduced voltages can substantially decrease energy/operation of an Intel processor [4]. However, finding optimum operating points without inducing timing errors is a challenge.

Reducing the supply voltage of FFT processors is a straightforward solution to reduce energy consumption. Due to the quadratic dependence of dynamic energy and linear dependence of static energy on voltage in digital circuits, even minor voltage reductions can result in substantial energy savings. Despite this, to reduce testing and characterization costs, manufacturers add surplus margin voltages [6] to the minimum operable voltage of their chips. This is to ensure that all delivered chips fulfill the stated performance specifications in the given range of environmental conditions [6], [7]. The increased power dissipation due to those margins can be as high as 30% according to studies on commercial platforms [6]. Furthermore, reducing the supply voltage of digital circuits down to near- and sub-threshold voltage levels promises a multifold improvement in efficiency [8], as depicted in Fig. 1.

Nonetheless, when a circuit is overclocked or the supply voltage is reduced, the probability of timing errors increases. This is due to Process, Voltage, and Temperature (PVT) variations, which make static timing analysis difficult [7].

To enable reduced voltage operation, multiple techniques have been proposed to monitor faults in computations during the voltage adjustment. Most of the proposed circuit-level techniques for mitigation of PVT variations at reduced voltages are complex and demand significant effort in the design and verification stages [9]. Those are not applicable for either commodity processors or commercial IP blocks. In contrast, in the current contribution a low-overhead approach to enable safe voltage reduction (or overclocking) of FFT processors is employed. It can even be implemented in software.

This work was supported by 6G Flagship research programme under Academy of Finland Grant 318927.

Mehdi Safarpour and Olli Silvén are with the Center for Machine Vision and Signal Analysis, University of Oulu, Oulu, Finland (e-mail: firstname.lastname@oulu.fi).

This article has been accepted for publication in IEEE Embedded Systems Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LES.2022.3212776

II. Fault Detection Techniques

At reduced voltages, fault detection mechanisms can be employed in finding the optimum operating point. That is the lowest voltage with the highest clock frequency that provides the correct results.

The well-known Error Correction Codes (ECC) are extensively used for detecting errors in data in telecommunication and storage applications and have been successfully leveraged for voltage scaling of memories and caches [10]. However, ECCs can't be utilized for detecting errors in computing.

In the following, a brief review of existing techniques for optimizing the operating voltage of computing blocks is given.

A. Hardware Based Schemes

Hardware based schemes consist of circuit-level augmentations or modifications. These are included at design time to estimate the timing slack or to detect timing errors in situ [11].

The simplest set of approaches for monitoring the timing errors are slack estimation methods. Those work by adding logic delay chains that mimic the longest delay paths of the original design. As a "canary in cage" they are supposed to generate errors before the main delay path does when the voltage is being reduced. Hence, the mimicking circuits provide guidance for optimal tuning of the operating voltage. Unfortunately, local temperature and process variations, circuit aging, and cross-coupled noise, impose a strong trade-off on energy conservation and reliability in this approach [7].

Similarly, offline calibration approaches [12] determine the minimum voltage margin of a given device by intermittent offline measurements and update a calibration table. Though commercially adopted, this approach has limited energy efficiency contribution as the offline calibration is susceptible to variable operating environment and circuit aging [13].

Instead of emulating the longest delay paths, the timing errors can be detected within the delay path, in situ, by using Timing Error Detection (TED) systems. These schemes are based on Error Detection Sequence (EDS) circuits in which a secondary register, along with the primary one, sample output of the combinational logic path within a pipeline stage. The secondary register samples with a slightly delayed clock [14]. A difference in the outputs between the primary and the secondary registers reveals late arrival signal and hence a timing error. Wang et. al. [15], utilized a TED system in their FFT to achieve near-threshold voltage operation. The design and testing complexity of slack estimation and TED systems discourage their adoption for low-voltage purposes. Recently, automated tools were introduced by Jiang et al. [13], to enable FPGAs to benefit from the approach. To the best knowledge of the authors, due to high design costs, very few successful commercial TED based designs have made their way to the market [16].

B. Algorithmic schemes

An alternative to hardware based error detection methods is the utilization of algorithm based approaches. Replication schemes such as Dual Modular Redundancy (DMR) cost too much in overhead which eats away the gains from voltage down-scaling. A naive approach is simply exercising the computing unit, e.g., an FFT accelerator, with prior known inputs for verification within certain time intervals. Again, the overheads and high rate of undetected errors can make this approach unattractive [17], [3].

Huang and Abraham introduced Algorithm Based Fault Tolerance (ABFT) for matrix operations [18]. ABFT has been demonstrated to be robust and attractive with an overhead ratio of $O(1/N)$, with N being the size of the matrix. FFT methods can also benefit from ABFT, however, that necessitates structural modifications to the FFT network [19]. Similarly, other fault detection schemes proposed in the literature, e.g., [19], require circuit or algorithm level modification within the FFT network and add non-trivial overheads to design time and complexity.

In this letter, we propose to employ a simple yet effective error detection solution specific to the Fourier transform for enabling reliable low-voltage operation of FFT hardware. An inherent property of Fourier transform is utilized for the detection of computing faults by leveraging the technique proposed in [3] into low-voltage operation. The outcome is better energy efficiency for any given design without architectural or circuit-level modifications to the FFT network.

III. Proposed Solution for FFT methods

As the guiding indicator to find the optimum operating points through computing fault detection we propose exploiting Parseval's identity. It is an intrinsic property of Fourier transform.

Using this approach, fault detection can take place on-the-fly by observing the inputs and outputs without modification of the FFT accelerator itself. This is important when considering the design cost and time, and allows using the approach with off-the-shelf IP cores or FFT accelerators in existing SoCs.

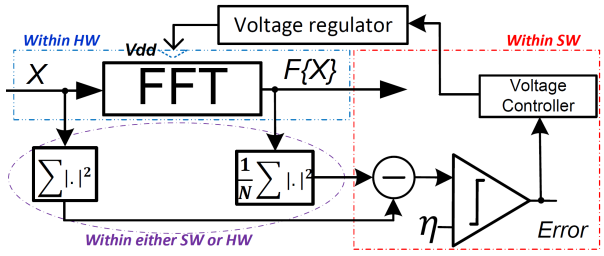

Fig. 2. Operating voltage of the FFT accelerator is adjusted according to the errors detected by using Parseval's identity. "η" is the error threshold.

A. Discrete Fourier Transform

The DFT of an N-point sequence $x(n)$, is defined as,

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \qquad k = 0, 1, \cdots, N-1 \qquad (1)$$

where $W_N = e^{-j(2\pi/N)}$.

The computational complexity of the above formulation is $O(N^2)$. FFT algorithms optimize the complexity of the DFT down to $O(N \log N)$. The DFT can be computed with any FFT algorithm and format [3].

Earlier in [3] Parseval's identity was proposed as a lightweight error detection approach in the FFT network. In our case, Parseval's identity is used to enable reduced voltage operation. It states that the sum of the squares of an input vector is equal to the weighted sum of the squares of its DFT as described by Eq. 2. Comparison of the numerical results from the left and right gives a fault detection approach that is somewhat similar to the checksum based error detection in the matrix operations of Huang and Abraham [20].

$$\sum_{k=0}^{N-1} |x[n]|^2 = \frac{1}{N} \sum_{k}^{N-1} |X[k]|^2 \tag{2}$$

The approach is depicted in Fig. 2. The FFT core computes the discrete Fourier transform. Meanwhile, as data enters and exits the accelerator, the energies of the inputs and the respective outputs are calculated and compared either in hardware or by software to check whether Parseval's identity holds. The host adjusts the operating voltage of the accelerator based on the observed errors.

B. Error Coverage

Without sufficient error detection capability, the reliability of the results obtained at reduced voltages could be compromised. In particular, the risk is the undetected faults [20].

The error detection performance of Parseval's identity based method strongly depends on the bit-width of the implemented FFT. The relation has been analyzed in [3] where the lower bound of fault coverage for 16-bit FFT is shown to be higher than 99.6%. For 32-bit FFT the coverage is close to 100% [3].

C. Overheads

The computational overhead translate into energy consumption overhead. Assuming a complex input vector size of N, extra $2N$ multiplications and $2N-1$ summations are needed for calculating the sum of squares for input and output sequences of the FFT. This results in an overhead complexity ratio of $O(1/\log_2(N))$ as analyzed in [3]. The overhead decline more moderately with respect to N when compared to the ratio $O(1/N)$ of Huang and Abraham's ABFT method for matrix multiplication [20]. However, the rate of reduction is fast enough to render the overhead negligible for large FFTs.

IV. Experimentation and Results

To investigate the efficiency of the proposed solution, a Xilinx Zynq SoC (part number XC7Z020-CLG484-1) with onboard voltage adjustment capability was utilized. An FFT processor was implemented on the Programmable Logic (PL) side of the SoC. Firmware was written for an ARM Cortex core on the Processing Subsystem (PS) side that employs the PL side as an FFT accelerator with a clock frequency of 50 MHz. It should be noted that near-threshold operation experimentation on this platform is not feasible at voltages below $\approx 0.54\,V$ due to crashing, regardless of clock frequency.

The setup is depicted in Fig. 3. The voltage of the PL was gradually reduced by controlling the on-board UCD7242 voltage regulators through Power Management BUS (PMBUS)

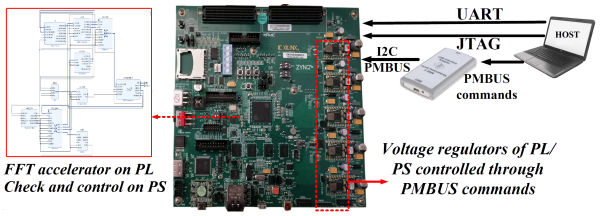

Fig. 3. Experimental setup using a Zynq SoC [21].

commands transmitted via I^2C bus from the PC. The voltage rails on the PL side of the SoC, i.e., VCCINT, VCCBRAM, and VCCAUX were aggressively scaled. In an application system, the processor that performs error checking could control the voltages.

A. Power savings

The energy savings depend on the overhead of the error checking scheme and its implementation. Figure 4 shows the power consumption of the PL, and the actual and detected error rates while the voltage is reduced. All three voltage rails were separately down-scaled until the point-of-first-failure (PoFF) was observed. As is seen in Fig. 4, there is a significant voltage margin, that accounts for about half of the power consumption within the FPGA. The margin is safely removed by feedback received from the error detection approach, saving $\approx 43\%$ of power, overall.

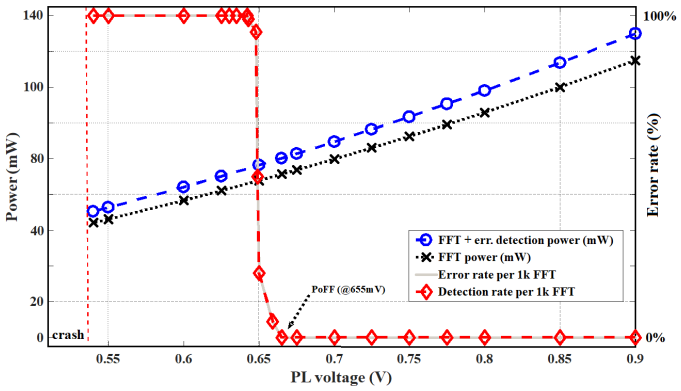

Fig. 4. Power consumption, overhead, and error detection of the approach.

The overhead when checking every FFT result for errors is depicted in Fig. 4. As shown in Fig. 2 error checking can be implemented either in hardware or/and software.

In the case of hardware based error checking of a 1024-point FFT, the signal energy computing blocks add ≈ 12 mW at the nominal voltage. The summary of the resource utilization is given in Table I. If error checking is fully implemented in software and is performed for every FFT result, it takes $\approx 8\%$ of total cycles amounting to ≈ 31 mW of PS power.

TABLE I
The Resource Utilization Estimates for a 1024-Point Design

	Power (@1.0V)	LUT	DSP	FF	BRAM
FFT core	145 mW	7845	18	11915	37
Error check	12 mW	873	5	548	8

TABLE II
COMPARISON WITH OTHER WORKS

Ref.	Application	Online	Integration	Overheads	Coverage
[12]	General	No	HW/Pre. fab	$\approx 0\%^a$	N/A
[14]	General	Yes	HW/ Pre. fab	1.6%	$>96.7\%$
[13]	General	Yes	HW/Pre. fab	0.4%-1.5%	$>98.5\%$
LoFFT	FFT	Yes	SW/Post fab	$<8\%^b$	$\approx 100\%^c$

[a] requires disruptive intermittent re-calibration
[b] for 1024-point FFT
[c] for bit-widths $>$ 16 bits [3].

B. Error detection

In Fig. 4 the PoFF is flagged by the error detection approach, and all errors are reported when the voltage is further reduced. Due to the low undetected error rate in experimentation, error detection was also investigated numerically in a MATLAB model similar to [3]. The simulation model adopted from [17] was utilized. In simulations the errors were injected as random uniform noise into a random set of frequency bins [3]. The fault alarm is triggered when the error is larger than the error threshold, η, complying with [3]. The approach detects errors as low as $\eta = 2^{-16}$ with accuracy very close to 100% as previously shown by [3], thus we do not include the data.

C. Comparison and Discussion

Table II presents a comparison of the proposed approach with the state-of-the-art. To the best knowledge of the authors, this is the first work on software based error-detection to enable FFT hardware to operate at reduced voltages. Moreover, online error detection supports adaptation to temperature dependencies, unlike the offline approaches [12]. The proposed approach can be used even when the SoC has been fabricated, provided that the FFT accelerator resides within an independent voltage domain.

Implementing the proposed approach in hardware enables buying energy efficiency with moderately increased die real estate. Even then the approach is simple and straightforward to implement in comparison to the previous low-voltage approaches, e.g., [9], [14], [12] that require substantial circuit-level modifications or cell characterization.

A simple alternative is to decimate the input and compare it against the DC component of the DFT [19]. Unfortunately, that approach can't detect errors in the multiplication circuitry of the FFT network and suffers from low error coverage [19]. The ABFT approach for matrix operations [20] would be inefficient as it requires performing DFT through matrix-vector multiplication, resulting in the complexity of $O(N^2)$ [19].

V. Summary

The current contribution demonstrates the efficacy of a straightforward fault detection technique based on Parseval's identity for enabling the low-voltage operation of FFT processors. The approach was demonstrated on an FPGA SoC with aggressive voltage down-scaling regimes, without requiring modifications to either the FFT processor, i.e., the netlist, the HDL code, or the FFT algorithm. The demonstrated energy efficiency gain at reduced voltage was $\approx 43\%$.

VI. Acknowledgement

The authors appreciate the feedback they received from Dr. Miguel Bordallo Lopez and Mr. Joseph Russell.

References

[1] U. F. Mohammad and M. Almekkawy, "A substitution of convolutional layers by fft layers-a low computational cost version," in *2021 IEEE International Ultrasonics Symposium (IUS)*. IEEE, 2021, pp. 1–3.
[2] M. Mahdavi, O. Edfors, V. Öwall, and L. Liu, "A low latency fft/ifft architecture for massive mimo systems utilizing ofdm guard bands," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 7, pp. 2763–2774, 2019.
[3] S.-J. Wang and N. K. Jha, "Algorithm-based fault tolerance for fft networks," *IEEE Transactions on Computers*, vol. 43, no. 7, pp. 849–854, 1994.
[4] S. Jain, S. Khare, S. Yada, V. Ambili, P. Salihundam, S. Ramani, S. Muthukumar, M. Srinivasan, A. Kumar, S. K. Gb, *et al.*, "A 280mv-to-1.2 v wide-operating-range ia-32 processor in 32nm cmos," in *2012 IEEE international solid-state circuits conference*. IEEE, 2012, pp. 66–68.
[5] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of computation*, vol. 19, no. 90, pp. 297–301, 1965.
[6] P. Koutsovasilis, C. Antonopoulos, N. Bellas, S. Lalis, G. Papadimitriou, A. Chatzidimitriou, and D. Gizopoulos, "The impact of cpu voltage margins on power-constrained execution," *IEEE Transactions on Sustainable Computing*, pp. 1–1, 2020.
[7] C. Krishna, "Global voltage scaling across multiple cores for real-time workloads," *IEEE Embedded Systems Letters*, 2022.
[8] R. Uytterhoeven and W. Dehaene, "Design margin reduction through completion detection in a 28-nm near-threshold dsp processor," *IEEE Journal of Solid-State Circuits*, vol. 57, no. 2, pp. 651–660, 2021.
[9] A. Wang and A. Chandrakasan, "A 180-mv subthreshold fft processor using a minimum energy design methodology," *IEEE Journal of solid-state circuits*, vol. 40, no. 1, pp. 310–319, 2005.
[10] G. Yalcin, E. Islek, O. Tozlu, P. Reviriego, A. Cristal, O. S. Unsal, and O. Ergin, "Exploiting a fast and simple ecc for scaling supply voltage in level-1 caches," in *2014 IEEE 20th International On-Line Testing Symposium (IOLTS)*. IEEE, 2014, pp. 1–6.
[11] N. D. Gundi, T. Shabanian, P. Basu, P. Pandey, S. Roy, and K. Chakraborty, "Effort: A comprehensive technique to tackle timing violations and improve energy efficiency of near-threshold tensor processing units," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 10, pp. 1790–1799, 2021.
[12] S. Zhao, I. Ahmed, C. Lamoureux, A. Lotfi, V. Betz, and O. Trescases, "Robust self-calibrated dynamic voltage scaling in fpgas with thermal and ir-drop compensation," *IEEE Transactions on Power Electronics*, vol. 33, no. 10, pp. 8500–8511, 2017.
[13] W. Jiang, H. Yu, H. Zhang, Y. Shu, R. Li, J. Chen, and Y. Ha, "Fodm: A framework for accurate online delay measurement supporting all timing paths in fpga," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 30, no. 4, pp. 502–514, 2022.
[14] K. Maragos, G. Lentaris, and D. Soudris, "A pvt-aware voltage scaling method for energy efficient fpgas," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2021, pp. 1–5.
[15] A. Wang and A. P. Chandrakasan, "Energy-aware architectures for a real-valued fft implementation," in *Proceedings of the 2003 international symposium on low power electronics and design*, 2003, pp. 360–365.
[16] J. Nunez-Yanez and N. Howard, "Energy-efficient neural networks with near-threshold processors and hardware accelerators," *Journal of Systems Architecture*, vol. 116, p. 102062, 2021.
[17] M. Safarpour, R. Inanlou, and O. Silvén, "Algorithm level error detection in low voltage systolic array," *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2021.
[18] D. Filippas, N. Margomenos, N. Mitianoudis, C. Nicopoulos, and G. Dimitrakopoulos, "Low-cost online convolution checksum checker," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2021.
[19] J.-Y. Jou and J. A. Abraham, "Fault-tolerant fft networks," *IEEE Transactions on Computers*, vol. 37, no. 5, pp. 548–561, 1988.
[20] K.-H. Huang and J. A. Abraham, "Algorithm-based fault tolerance for matrix operations," *IEEE transactions on computers*, vol. 100, no. 6, pp. 518–528, 1984.
[21] M. Safarpour, L. Xun, G. V. Merrett, and O. Silvén, "A high-level approach for energy efficiency improvement of fpgas by voltage trimming," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2021.