Energy Optimizations for FPGA-based 2-D FFT Architecture

Ren Chen and Viktor K. Prasanna
Ming Hsieh Department of Electrical Engineering
University of Southern California
Los Angeles, USA 90089
{renchen,prasanna}@usc.edu

***Abstract*—Row-column algorithm is commonly used for 2-D FFT implementation on FPGA. However, in this algorithm, the strided memory access to external memory such as DRAM introduces significant delay for DRAM row activation, thus resulting in high DRAM energy and a significant amount of FPGA device energy consumed in idle state. In this paper, to optimize energy consumption of the 2-D FFT architecture, we employ an FPGA-based 1-D FFT kernel supporting processing streaming data to fully utilize the available bandwidth offered by the external memory , and balance the I/O bandwidth between the DRAM and FPGA to minimize the FPGA idle time. Furthermore, to avoid time consuming DRAM row activation, we decompose the required transposition by column-wise FFTs into smaller size problems, thus enabling on-chip local transposition which could be performed by the customized data permutation unit used in the 1-D FFT kernel. Compared with the baseline 2-D FFT architecture, the optimized architecture achieves 3.9x, 4.2x and 4.5x improvement in energy efficiency for 1024×1024, 4096×4096 and 8192×8192 points 2-D FFTs, respectively. We also estimate the *peak energy efficiency* of the FPGA-based 2-D FFT architecture. Our estimation shows that our optimized 2-D FFT Kernel can achieve $8.06 \sim 8.31$ GFLOPS/W for various 2-D FFTs, ie., up to 62% of the *peak energy efficiency* of 2-D FFT architecture on FPGA.**

I. Introduction

Fast Fourier Transform (FFT) is one of the frequently used kernels in a variety of image and digital signal processing applications [1], [2], [3], [4]. Particularly, the two-dimensional (2-D) FFT is extensively used in a wide range of imaging applications [5], [6], [7], [8]. The continuously increasing input image size results in high throughput requirement. To meet the throughput requirement, maximizing performance under energy dissipation constraints is a critical issue.

In imaging applications using 2-D FFT, the system capability to support non-stop processing on continuous image streams is critical to achieve high throughput performance. FPGA-based system provides such capability benefiting from its abundant computation and routing resource. There have been several prior research works on FPGA-based 2D FFT design using row-column algorithm [7], [9]. In those works, DRAM is commonly used as the external memory to store the image data, and on-chip 1-D FFT kernel is implemented on FPGA to process the data intensive computations. The energy consumption of a 2-D FFT implementation is not only determined by the power performance of the system components including the external memory and the on-chip 1-D FFT module, but also affected by the throughput disparity between the system components. The throughput disparity leads to extra idle time of one of the components, and thus lowering the entire system throughput. A throughput balanced architecture minimizes the throughput disparity between the system components, thus reducing the energy consumed in idle state unnecessarily. The run-time throughput of DRAM is highly affected by the page hit rate. Page hit rate represents the percentage of times of access to an open row of the external memory. In row-column algorithm, the strided memory access to DRAM results in a page hit rate reduction, hence introducing significant delay for DRAM row activation and lowering the entire system throughput. Besides, as DRAM row activation leads to a higher average activate current, power consumption per DRAM access increases correspondingly [10].

In this paper, to optimize the energy consumption of FPGA-based 2-D FFT design, we develop a throughput balanced architecture. We observe that the required data transposition on external memory is one of the key factors affecting the entire system throughput. To minimize the throughput disparity between the FPGA device and DRAM, we propose a data remapping method to decompose the data transposition into smaller size problems. In this way, the DRAM bandwidth utilization is effectively improved. Furthermore, to reduce the times of DRAM row activation, input data employed by several consecutive column-wise 1-D FFTs are prefetched onto local memory on FPGA. A permutation network is developed to permute the prefetched data to retrieve the original data layout. We redesigned the permutation network required by the 1-D FFT kernel so that it is also able to handle the small block-size data transposition locally on-chip. The redesigned permutation network largely reduces the extra hardware overhead, and also saves energy consumption. The contributions in this paper are summarized as follows:

1) An energy efficient throughput-oriented floating-point 2-D FFT implementation on FPGA (Section III-A).
2) Throughput balanced 2-D FFT architecture showing significant energy performance improvement (Section III-B).
3) A data remapping method leading to a high DRAM page hit rate (Section III-C).
4) Significant energy efficiency improvement compared with the baseline architecture (Section IV-D).

This work has been funded by DARPA under grant number HR0011-12-2-0023. The content, views and conclusions presented in this document do not necessarily reflect the position or the policy of DARPA.

978-1-4799-6233-4/14/$31.00

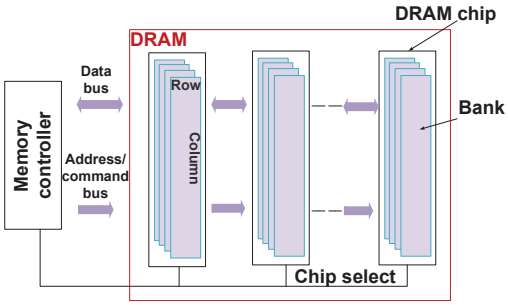


Fig. 1: The overview of DRAM organization

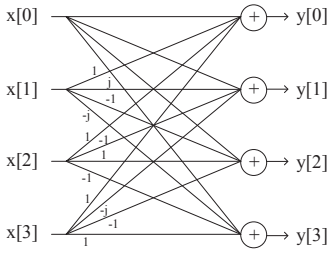


Fig. 2: Raidx-4 block

The rest of the paper is organized as follows. Section II covers the background and related work. Section III introduces the proposed architecture and its implementation on FPGA. Section III-C describes the proposed data remapping method for DRAM access. Section IV presents experimental results and analysis. Section V concludes the paper.

II. Background and Related Work

As the well-known simplest multidimensional FFT algorithm, the row-column algorithm has been commonly used to implement 2-D FFT by performing a sequence of 1-D FFTs. In this algorithm, input elements hold by an $N \times N$ array are stored in row-major order in the external memory such as DRAM. First, row-wise 1-D FFT is applied to each of the N rows and the computation results also represented using an $N \times N$ array are written back to the external memory. Then column-wise 1-D FFT is applied to each of the N columns of the data array generated by the row-wise 1-D FFTs. Different from small size 1-D FFT, large input image size requires external memory such as DRAM to be employed.

Generally, a channel of DDR3 DRAM is organized into ranks, and each rank consists of several DRAM chips [11]. The DRAM organization is shown in Fig. 1. A DRAM chip is composed of several banks. Each bank is composed of a two dimensional matrix of locations. At each addressable location ([Bank, Row, Column]), a fixed number of data bits (usually 8-bit) are located. A row of a bank needs to be activated before a read/write operation at one of its locations. The data given in [11] shows that the minimum activate-to-activate time for the same bank is 48.75 ns, however, this time for two different banks is 7.5 ns. An operation on an active row is defined as page-hit access. Page hit rate represents the percentage of page-hit access to the external memory during the run-time.

One major issue in the implementation of the 2-D FFT architecture is the considerable delay caused by DRAM row activation which are mainly introduced by the strided memory access in the column wise 1-D FFTs. To solve this problem, the authors in [7] propose a tiled data mapping method to improve the external memory bandwidth utilization. They logically divide the input $N \times N$ input array into $\frac{N}{k} \times \frac{N}{k}$ tiles and map the elements in each tile to consecutive memory locations. They conclude that the DRAM bandwidth utilization is maximized when the size of each tile is set to be the size of the DRAM row buffer. However, this solution introduces non-trivial on-chip hardware resource cost for local transposition. Also energy consumption is not considered. A 2-D FFT architecture achieving high throughput performance with minimal row activation cost is presented in [6]. In this work, the authors propose a 2-D decomposition algorithm which enables local 2-D FFT on sub-blocks. In this way, the times of DRAM row activation is minimized. However, extra stages including row-wise and column-wise data exchange are introduced, and the total number of operations are increased from $2N^2 \log N$ to $2N^2(1 + \log N)$. Thus both the delay and energy will be increased considerably. Vector radix 2-D FFT in [8] presents a general structure theorem to construct a multi-dimensional FFT architecture by decomposing a large size problem into small size 2-D FFTs. Their decomposition method effectively reduce the number of multiplications, however, the external memory activation issue is not considered in this paper. There also have been other implementations of single or multi-dimensional FFTs on various platforms [12], [13], [14], however, none provide any solution to solve memory row activation problem.

Those prior research works achieved high throughput by efficiently utilize DRAM maximum bandwidth, while energy was not considered. In this paper, we explored optimizations to maximize the performance of 2-D FFT architecture under energy consumption constraint. On the basis of the low power implementation of the throughput-oriented 1-D FFT kernel, we balance the I/O bandwidth between the on-chip and the off-chip to minimize the energy consumed in idle state by the system components. Furthermore, we decompose the transposition required by column-wise 1-D FFTs into smaller size problems and redesign the on-chip permutation network employed by 1-D FFT kernel to take the responsibility of transposition. In this way, both the latency of one time 2-D FFT computation and the average power consumption of the system is reduced significantly.

III. Design and Implementation of 2-D FFT Architecture

A. 1-D FFT Kernel

An N-point (floating-point) 1-D FFT kernel is implemented by concatenating several basic components including radix block, data path permutation (DAP) unit, and twiddle factor computation (TWC) unit. The design of each architecture component relies on the FFT algorithm in use. Implementation details of those components will be introduced next. We applied several energy optimizations discussed in [15], [16] onto the design components to reduce their energy consumption.

1) Radix block: The radix block is used to perform a butterfly computation on some input samples. For example, the

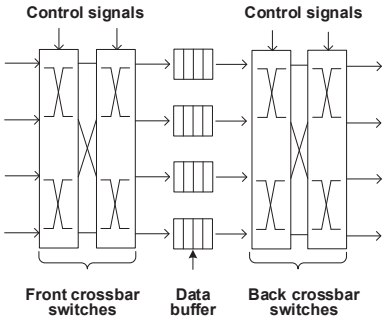


Fig. 3: DAP unit for Radix-4 FFT

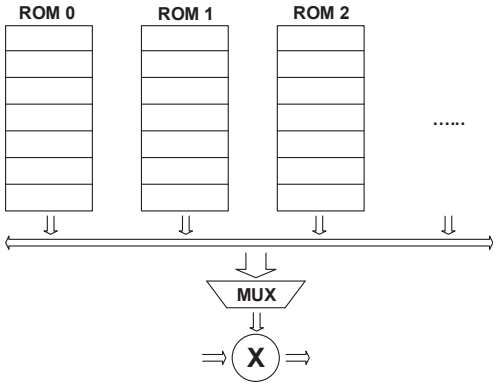


Fig. 4: Hardware structure of TWC unit

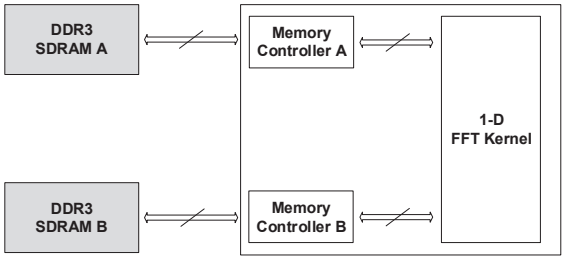


Fig. 5: The baseline 2-D FFT architecture with DRAM as external image memory

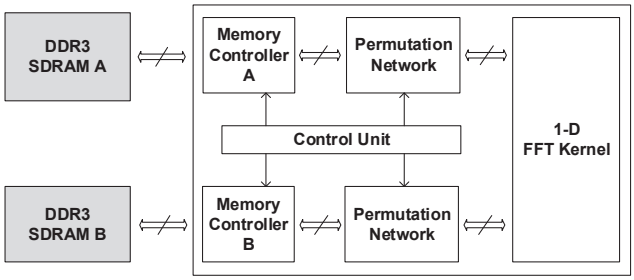


Fig. 6: The optimized 2-D FFT architecture with DRAM as external image memory

radix block for radix-4 FFT takes four input samples, performs the butterfly computation and then generates four results in parallel. Each radix block is composed of complex adders and subtractors. The structure of a radix block is determined by the FFT algorithm in use. Fig. 2 shows the structure of radix block for radix-4 FFT.

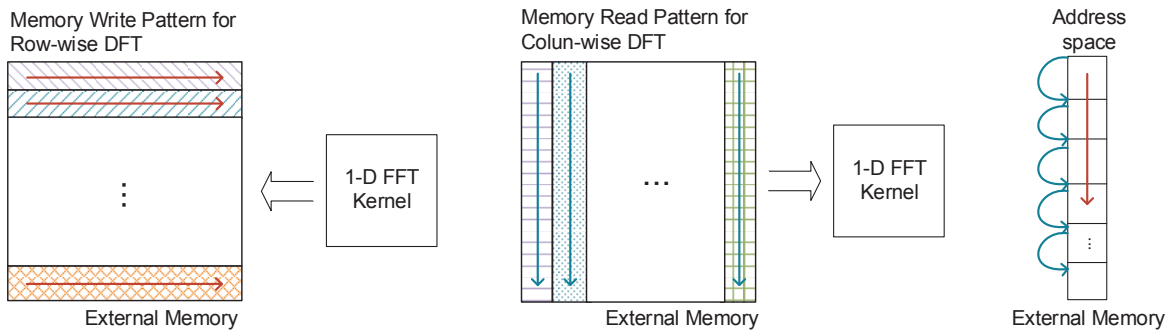
2) DAP Unit: DAP unit is used for data permutation between butterfly computation stages in FFT. A DAP unit is composed of demultiplexers and data buffers. In subsequent clock cycles, data from previous butterfly computation stage are first multiplexed and written into several data buffers. Each stored data element will be buffered with a certain number of clock cycles and then read out. Outputs from data buffers will also be multiplexed and fed into the next butterfly computation stage. Fig. 3 shows the DAP unit used for a radix-4 based FFT design. Each DAP unit consists of eight 1-to-4 demultiplexers and four data buffers. In each cycle, a data buffer may be read and written simultaneously on different addresses. The size of each data buffer depends on the ordinal number of its present butterfly computation stage and the FFT problem size. Note that each data element is a complex number including both its real part and imaginary part, hence the data width is 64 bit.

3) TWC Unit: A TWC unit consists of two parts: the TWC generation logic and the complex number multiplier. As shown in Fig. 4, the TWC generation logic includes several lookup tables (ROMs) for storing twiddle factor coefficients, where the data read addresses will be updated with the control signals. The size of each lookup table is determined by the ordinal number of its present butterfly computation stage and the FFT problem size. Each lookup table can be implemented using a BRAM or distributed RAM (dist. RAM) on FPGA [17]. Each complex number multiplier consists of four real number multipliers and two real number adders/subtractors.

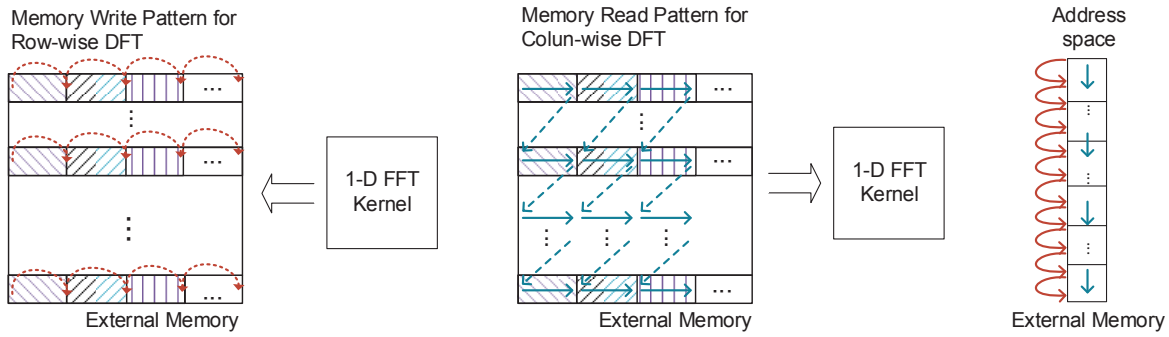
B. Throughput balanced 2D FFT Architecture

A naïve baseline architecture is shown in Fig. 5. The external memory is composed of several DRAM chips, each of which has 16-bit output data width. The number of DRAM chips is determined by the memory bandwidth required by the FFT processors on FPGA. In the baseline architecture, with the knowledge that the 1-D FFT kernel has a throughput of 3.2 GB/s at 100 MHz, a DRAM chip with 16 data pins running at 800 MHz can be employed to meet the memory bandwidth requirement [11]. Note that more memory bandwidth could be obtained by using multiple DRAM chips in parallel, however, more memory energy will be consumed correspondingly. During the system run-time, unused DRAM chips will be set to precharge power-down mode in our experiments.

The 2-D images are stored on the DRAM in row-major order. The row-wise 1-D FFT will first be performed, with the image data read from the DRAM in sequence row-by-row. The data results of row-wise 1-D FFTs will also be stored onto the DRAM in sequential. Then the column-wise 1-D FFTs will be processed by reading data from the DRAM in column-major order. As the column-wise 1-D FFTs need to wait for the outputs from the row-wise 1D FFTs, the performance of the 2-D FFT is limited by the bandwidth of external memory. Also, the column-wise memory accesses on DRAM are time consuming due to the overhead introduced by activation on different DRAM rows. In the baseline architecture, we employ



(a) Before data remapping



(b) After data remapping

Fig. 7: DRAM access patterns in 2-D FFT

a Radix-4 based 1-D FFT kernel III-A without any optimization and set the operating frequency as 100 MHz. Maximum frequency is not optimized as energy optimization is our focus in this work.

The proposed 2-D FFT architecture is shown in Fig. 6, in which a controlling unit (CU) and a permutation network are introduced. The CU is developed for accessing DRAM out-of-order (with small strided addresses). The permutation network is proposed for time reordering. In baseline architecture, when performing column-wise 1-D FFTs, memory address is increased with a stride equals to FFT problem size N after each memory access. However, DRAM chips require a minimum activate-to-activate delay when successively accessing two rows in the same bank. This delay results in a huge decline in external memory bandwidth utilization, thus the entire system throughput is impaired. In the optimized architecture, the controlling unit is responsible for updating memory address dynamically so that data results of row-wise 1-D FFTs could be mapped onto DRAM using the optimal data layout. Through this data remapping, DRAM row activation will be only needed after several successive accesses on the same row rather than executed each time after a DRAM access. Thus the impact of DRAM row activations on the entire system throughput will almost be minimized. Furthermore, to reduce the times of DRAM row activation, data inputs of several consecutively performed column-wise 1-D FFTs will be moved from DRAM to local memory together, without waiting for the completion of the current executed 1-D FFT. For this purpose, the permutation network is necessary so that these prefetched data could be reordered to be processed by column-wise 1-D FFTs.

C. Data remapping on DRAM

In the baseline, data are written sequentially by row-wise 1-D FFTs. Then those results are read using strided addresses by the column-wise 1-D FFTs. After data remapping, the data of the row-wise 1-D FFTs are also written by using strided addresses, and the results are read block-by-block by the column-wise 1-D FFT. The permutation network will be employed for permuting the data in those blocks locally. The permutation network is actually implemented by DAP units, which are responsible for data permutation between 1-D FFT butterfly computation stages. We combine the permutation network with the input DAP unit and the output DAP unit in the 1-D FFT kernel for resource and power consumption reduction.

Instead of writing the results sequentially to DRAM in row-wise FFTs, we remap the results of row-wise FFTs when writing them onto DRAM. Fig. 7a shows the DRAM access pattern before data remapping in baseline. For row-wise 1-D FFTs, theirs results are written row-by-row sequentially onto DRAM. As a result, strided memory access is performed by column 1-D FFTs. Fig. 7b shows the DRAM access patterns after data remapping. The results of row-wise 1-D FFTs are written onto DRAM with a stride, which is determined by the minimum activate-to-activate delay when accessing the same bank and the DRAM operating frequency. Then DRAM needs to be accessed by column-wise 1-D FFTs block by block. A column-wise 1-D FFT will employ data elements from blocks in the same color.

IV. Experimental Results and Analysis

In this section, we compare the energy efficiency of the optimized 2-D FFT architecture and the baseline architecture with a sustained throughput of at least 3.2 GB/s for the 1024×1024, 4096×4096 and 8192×8192 points FFTs. Section IV-A details the experimental setup. Section IV-B introduces the metric of peak energy efficiency. Section IV-C presents the results of optimized energy consumption for DRAM access. Section IV-D demonstrates the improvements in energy for the optimized 1-D FFT kernel and the improvements achieved using our optimized 2-D FFT architecture.

A. Experimental Setup

The experiments were conducted on the state-of-the-art FPGA-based MPSoC platform Zynq-7000 plus a 512 MB DDR3 memory. The FPGA device on Zynq-7000 is Xilinx Artix 7 [17]. The experiments were performed using Xilinx Vivado 2013.4 development tools including the Vivado Power Analysis tool used for determining the power dissipation [18]. The designs were verified by post place-and-route simulation. The input matrices for the simulation were randomly generated and had an average switching activity of 50%. All of the reported results are post place-and-route simulation results. We used the VCD (Value Change Dump Format) file as input to the Xilinx Vivado Power Analyzer to produce accurate power dissipation estimation.

B. Peak Energy Efficiency

The energy efficiency of any 2-D FFT design is upper bounded by the inherent peak performance of the platform. This depends on the target device and the IP cores used to perform the arithmetic operations. We measure this *Peak Energy Efficiency* by using an minimal architecture to depict the ideal conditions. First, we assume the DRAM row activation will not affect the system throughput. Second, we ignore the overhead such as I/O and routing (interconnection on FPGA) that may be employed by an FPGA-based implementation. This minimal architecture consists of only multipliers, adders, buffers and DRAM as the basic components. Each computation unit has its own input registers for pipelining. Using the proposed experimental setup, we estimated the power consumption of the adder and the multiplier from Xilinx LogiCORE IP Cores [17]. We compute the *Peak Energy Efficiency* as 13.4 GFLOPS/W for Radix-4 based $N \times N$-point 2-D FFT. Note that *Peak Energy Efficiency* is independent on interconnection and DRAM page hit rate, and as the multiplier and adder cores improve, we can expect a corresponding increase in the *Peak Energy Efficiency*. We use the *Peak Energy Efficiency* as an upper bound on the performance of the chosen FFT algorithm. We also compare the sustained performance of our designs against this bound in Section IV-D.

C. Energy Consumption in the External Memory

Before evaluating the energy consumption of the entire design, we separately estimate the DRAM power consumption for both the baseline and the optimized architecture using DRAM power calculator [11]. The metric is defined as energy per read. For energy evaluation, we first obtain the operation time in seconds by 2-D FFT, and then estimate the energy consumed by the DRAM using DRAM power calculator. Table I shows the energy performance of the external memory before and after the proposed optimization. There is no much energy performance difference between the baseline architecture and the optimized architecture regarding DRAM access by row-wise 1-D FFTs. The reason for that is the system throughput is almost not affected by row-wise 1-D FFTs in the optimized architecture. From the Table I, it shows that more energy is consumed by per read for column-wise FFT with a larger problem size. Through the proposed optimization, the energy per read by DRAM is reduced by 48.6%, 57.8%, and 59.2% for 1024×1024, 4096×4096 and 8192×8192 size 2-D FFTs respectively.

TABLE I: DRAM energy performance comparison

	1024×1024 2-D FFT	4096×4096 2-D FFT	8192×8192 2-D FFT
Energy per read for column-wise FFT (Baseline)	3.43 nJ	5.48 nJ	6.42 nJ
Energy per read for column-wise FFT (Optimized)	1.76 nJ	2.31 nJ	2.62 nJ
Reduction percentage (Energy per read)	48.6%	57.8%	59.2%

D. Performance Comparison

We conducted performance comparison using two metrics. One metric is defined as energy per point, which refers to a complex number computed by 2-D FFT. Another performance metric which is energy efficiency, defined as GFLOPS/W, is also employed in this performance comparison. In order to give a thorough view of the energy efficiency of the entire 2-D FFT architecture, we evaluate the static power dissipation of the architecture and calculate the energy efficiency including the static power dissipation. According to our power simulation results, the static power of the Artix-7 is 0.085W for all the optimized and baseline architectures, regardless of how many resources are used and the operating frequency.

We first separately evaluate the performance of the optimized 1-D FFT kernel using the chosen metrics. In the baseline architecture, none of the proposed optimizations discussed in Section III-A are applied. The energy efficiency of the baseline and the optimized 1-D FFT kernel on FPGA device is calculated and listed in Table II. The optimized architecture achieves 3.3x, 3.3x and 3.4x energy efficiency improvement in 1024, 4096 and 8192 points 1-D FFTs, respectively. Besides, the energy consumption per point is reduced by 70.3%, 69.7%, 67.6% for 1024, 4096 and 8192 points 1-D FFTs, respectively.

By considering the energy consumed by both the DRAM and the 1-D FFT kernel, we compare the performance of the two target 2-D FFT architectures using the defined metrics. Table III presents the energy performance comparison between the baseline 2-D FFT architecture and the optimized 2-D FFT architecture. It shows that the entire optimized 2-D FFT architecture achieves 8.31, 8.01 and 8.06 GFLOPS/W in energy efficiency for 1024×1024, 4096×4096 and 8192×8192 points 2-D FFT when considering both dynamic power and static

TABLE II: Energy performance (static power and dynamic power) of the optimized 1-D FFT kernel on FPGA

	Baseline architecture			Optimized architecture			
FFT size	Power (W)	Energy per point (nJoule)	Energy efficiency (GFLOPS/W)	Power (W)	Energy per point (nJoule)	Energy efficiency (GFLOPS/W)	Energy efficiency improvement
1024	2.56	6.4	5.9	0.71	1.9	19.6	3.3x
4096	3.42	8.6	5.4	1.04	2.6	17.9	3.3x
8192	4.31	10.8	4.7	1.39	3.5	15.8	3.4x

TABLE III: Energy performance (static power and dynamic power) of the optimized 2-D FFT architecture

	Baseline architecture			Optimized architecture			
FFT size	Power (W)	Energy per point (nJoule)	Energy efficiency (GFLOPS/W)	Power (W)	Energy per point (nJoule)	Energy efficiency (GFLOPS/W)	Energy efficiency improvement
1024×1024	7.28	18.2	2.09	1.83	4.6	8.31	3.9x
4096×4096	9.82	24.5	1.89	2.32	5.8	8.01	4.2x
8192×8192	11.28	28.2	1.79	2.51	6.3	8.06	4.5x

power. The energy consumption per point is reduced by 74.8%, 76.4.7%, 77.7% for 1024×1024, 4096×4096 and 8192×8192 points 2-D FFTs, respectively. Comparing the results of the energy consumption per point in the 1-D FFT kernel and the entire 2-D FFT architecture, we observe that the optimization for DRAM access makes a major contribution in the improvement of energy performance. Moreover, the sustained energy efficiency of the optimized 2-D FFT architecture achieves up to 62% of the *peak energy efficiency* (Section IV-B), which is an upper bound on the performance of the chosen FFT algorithm and FPGA device. Note that when calculating the *peak energy efficiency*, we ignored the interconnection power and DRAM row activation cost, which are generally among the major contributors to the energy consumption of the implementations on system including both FPGA and DRAM.

V. CONCLUSION

In this paper, we proposed several energy optimizations to obtain an energy efficient throughput-oriented 2-D FFT architecture on FPGA. The proposed architecture achieves high throughput by maximizing and balancing the bandwidth between the external memory and 1-D FFT kernel on FPGA. By combining the transposition required in the external memory and the on-chip permutation in 1-D FFT, the delay caused by memory row activation is highly reduced, thus saving significant amount of energy consumed by one time computation of 2-D FFT. The experimental results comparing with the baseline architecture show that our implementation outperforms in energy efficiency. In the future, we plan to build a design framework targeted at throughput-oriented signal processing kernels, which enables automatic energy optimizations addressing new IC technologies.

REFERENCES

[1] L. R. Rabiner, B. Gold, and C. K. Yuen, "Theory and application of digital signal processing," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 8, no. 2, pp. 146–146, 1978.

[2] G. Bi and E. Jones, "A pipelined FFT processor for word-sequential data," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 12, pp. 1982–1985, 1989.

[3] R. Chen, N. Park, and V. K. Prasanna, "High throughput energy efficient parallel FFT architecture on FPGAs," in *Proc. of IEEE International Conference on High Performance Extreme Computing (HPEC '13)*, 2013.

[4] 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.

[5] S. Langemeyer, P. Pirsch, and H. Blume, "Using sdrams for two-dimensional accesses of long $2^n \times 2^m$-point ffts and transposing," in *Proc. of International Conference on Embedded Computer Systems (SAMOS '11)*, July 2011, pp. 242–248.

[6] J. S. Kim, C.-L. Yu, L. Deng, S. Kestur, V. Narayanan, and C. Chakrabarti, "FPGA architecture for 2D Discrete Fourier Transform based on 2D decomposition for large-sized data," in *Proc. of IEEE Workshop on Signal Processing Systems*, Oct 2009, pp. 121–126.

[7] B. Akin, P. Milder, F. Franchetti, and J. Hoe, "Memory bandwidth efficient two-dimensional fast fourier transform algorithm and implementation for large problem sizes," in *Proc. of IEEE International Symposium on Field-Programmable Custom Computing Machines (FCCM '12),*, April 2012, pp. 188–191.

[8] H. Wu and F. Paoloni, "The structure of vector radix fast fourier transforms," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 9, pp. 1415–1424, Sep 1989.

[9] W. Wang, B. Duan, C. Zhang, P. Zhang, and N. Sun, "Accelerating 2d fft with non-power-of-two problem size on fpga," in *Proc. of IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig '10)*, Dec 2010, pp. 208–213.

[10] S. Liu, S. Memik, Y. Zhang, and S. Memik, "A power and temperature aware dram architecture," in *Proc. of ACM/IEEE International Conference on Design Automation Conference (DAC '08)*, June 2008, pp. 878–883.

[11] "DDR3 SDRAM System-Power Calculator," http://www.micron.com/-/media/Documents/Products/Power%20Calculator/DDR3_Power_Calc.XLSM.

[12] N. Govindaraju, B. Lloyd, Y. Dotsenko, B. Smith, and J. Manferdelli, "High performance discrete fourier transforms on graphics processors," in *Proc. of IEEE International Conference on Supercomputing (SC '08)*, Nov 2008, pp. 1–12.

[13] M. Frigo and S. Johnson, "Fftw: an adaptive software architecture for the fft," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '98)*, vol. 3, May 1998, pp. 1381–1384 vol.3.

[14] F. Franchetti, M. Puschel, Y. Voronenko, S. Chellappa, and J. Moura, "Discrete fourier transform on multicore," *IEEE Signal Processing Magazine*, vol. 26, no. 6, pp. 90–102, November 2009.

[15] R. Chen and V. Prasanna, "Energy-efficient architecture for stride permutation on streaming data," in *Proc. of IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig '13)*, Dec 2013, pp. 1–7.

[16] R. Chen, H. Le, and V. K. Prasanna, "Energy efficient parameterized FFT architecture," in *Proc. of IEEE International Conference on FPL*, 2013.

[17] "XST user guide for Virtex-6, Spartan-6, and 7 series devices," http://www.xilinx.com/support/documentation.

[18] "Vivado Design Suite User Guide: Design Flows Overview," http://www.xilinx.com/support/documentation/sw_manuals/xilinx2013_4/ug892-vivado-design-flows-overview.pdf.