SMUL-FFT: A Streaming Multiplierless Fast Fourier Transform

Seyed Hadi Mirfarshbafan, Sueda Taner, and Christoph Studer

***Abstract*—Beamspace processing is an emerging paradigm to reduce hardware complexity in all-digital millimeter-wave (mmWave) massive multiple-input multiple-output (MIMO) basestations. This approach exploits sparsity of mmWave channels but requires spatial discrete Fourier transforms (DFTs) across the antenna array, which must be performed at the baseband sampling rate. To mitigate the resulting DFT hardware implementation bottleneck, we propose a fully-unrolled Streaming MUltiplierLess (SMUL) fast Fourier Transform (FFT) engine that performs one transform per clock cycle. The proposed SMUL-FFT architecture avoids hardware multipliers by restricting the twiddle factors to a sum-of-powers-of-two, resulting in substantial power and area savings. Compared to state-of-the-art FFTs, our SMUL-FFT ASIC designs in 65 nm CMOS demonstrate more than 45% and 17% improvements in energy-efficiency and area-efficiency, respectively, without noticeably increasing the error-rate in mmWave massive MIMO systems.**

I. Introduction

Millimeter-wave (mmWave) communication [1], [2] promises significantly increased data-rates due to the availability of large contiguous frequency bands. Massive multiuser multiple-input multiple-output (MU-MIMO) [3] is a key technology to combat the high path loss of mmWave propagation [2] while enabling simultaneous communication with multiple user equipments (UEs) in the same frequency band. The higher baseband sampling rates needed to support larger bandwidths at mmWave frequencies, combined with the large number of antennas in massive MU-MIMO, result in new challenges for analog and digital hardware design.

A. Fast Fourier Transforms for Beamspace Processing

MmWave channels typically comprise only a few dominant propagation paths [1], [2], making them sparse in the beamspace domain [4]–[9]. Beamspace processing exploits this sparsity to reduce the computational complexity of baseband processing [10]–[12]. This approach, which is described in detail in Section II-A, calls for spatial discrete Fourier transforms (DFTs) operated at the baseband sampling rate in order to convert the received signals at the antenna array into the beamspace domain—in high-bandwidth mmWave communication systems, billions of spatial DFTs must be computed per second.

S. H. Mirfarshbafan, and C. Studer are with the Department of Information Technology and Electrical Engineering of ETH Zürich, Switzerland; e-mail: mirfarshbafan@iis.ee.ethz.ch and studer@ethz.ch.

Sueda Taner is with the School of Electrical and Computer Engineering at Cornell University, Ithaca, NY, USA; e-mail: st939@cornell.edu.

The work of SHM, ST, and CS was supported in part by ComSenTer, one of six centers in JUMP, an SRC program sponsored by DARPA, by an ETH Research Grant, and by the US National Science Foundation (NSF) under grants CNS-1717559 and ECCS-1824379.

The literature describes a plethora of results on efficient DFT implementations based on the well-known fast Fourier transform (FFT) [13]. FFT architectures generally fall into four categories [14]: (i) iterative architectures, which require the smallest area but result in the lowest throughput and highest latency, (ii) serial pipelined architectures, which can process one complex sample per clock cycle, (iii) parallel pipelined architectures, which process multiple complex samples per clock cycle, and (iv) fully-unrolled architectures, which achieve the highest throughput by processing a full vector every clock cycle. While the bulk of research on FFT designs focuses on serial and parallel pipelined architectures, see e.g., [15]–[18], fully-unrolled architectures attracted less attention but appear to be the most suitable for beamspace transforms—see Section III-B for a detailed discussion. We note that analog beamspace transforms have been proposed in [19], but recent studies have shown that all-digital architectures using digital transforms can be advantageous in practice [20], [21].

B. Contributions

We propose a fully-unrolled Streaming MUltiplierLess (SMUL) FFT architecture suitable for mmWave massive MU-MIMO beamspace transforms. We restrict the twiddle factors so that their real and imaginary parts have at most two nonzero digits in the canonical signed digit (CSD) representation, which enables the use of constant multipliers that comprise of at most one adder and two arithmetic shifts.[1] To further improve energy- and area-efficiency, we deploy a specialized fixed-point number scaling schedule and bitwidth growth profile. In addition, our MATLAB-based Verilog generator automatically produces SMUL-FFT designs for different system parameters. We provide ASIC implementation results in 65 nm CMOS and compare our designs to state-of-the-art Spiral-FFTs [26].

C. Notation

Boldface lowercase letters represent vectors and uppercase letters represent matrices. The transpose of a matrix $\mathbf{A}$, is denoted by $\mathbf{A}^{\mathrm{T}}$. For a vector $\mathbf{a}$, the kth entry is $a_k = [\mathbf{a}]_k$, and the element-wise pth power is $\mathbf{a}^{\circ p}$. The ℓ_2-norm of $\mathbf{a}$ is $\|\mathbf{a}\|_2$ and the Frobenius norm of $\mathbf{A}$ is $\|\mathbf{A}\|_F$. The $B \times B$ identity matrix is denoted by $\mathbf{I}_B$ and the unitary DFT matrix by $\mathbf{F}$.

II. Background

A. Beamspace Massive MU-MIMO Processing

We consider a mmWave massive MU-MIMO system as depicted in Figure 1, in which a basestation (BS) equipped

[1] While multiplierless FFT designs have been explored in the past [22]–[25], only serial or parallel pipelined architectures have been considered.

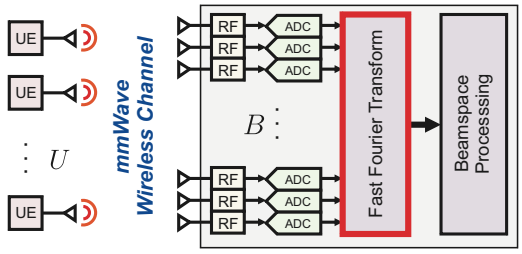


Fig. 1. Beamspace processing in mmWave massive MU-MIMO systems.

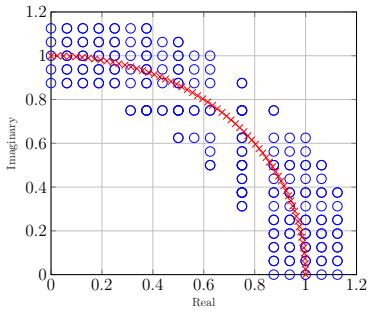


Fig. 2. Quantized twiddle factors: red cross markers are the exact twiddle factors, blue circles are the alphabet of complex numbers with a maximum of two nonzero digits in CSD representation of each real and imaginary part.

with a B-antenna uniform linear array (ULA), communicates with U single-antenna UEs. Let $\mathbf{H} \in \mathbb{C}^{B\times U}$ denote the channel matrix and $\mathbf{s} \in \mathbb{C}^U$ the vector of symbols transmitted by the U UEs. Then, the *antenna domain* received vector at the BS for a frequency-flat channel is modeled as $\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n}$, where $\mathbf{n} \in \mathbb{C}^B$ is the AWGN. By applying a DFT to the antenna-domain vector $\mathbf{y}$, we obtain the *beamspace domain* vector

$$\hat{\mathbf{y}} = \mathbf{F}\mathbf{y} = \hat{\mathbf{H}}\mathbf{s} + \hat{\mathbf{n}}, \tag{1}$$

where the matrix $\hat{\mathbf{H}} = \mathbf{F}\mathbf{H}$ and vector $\hat{\mathbf{n}} = \mathbf{F}\mathbf{n}$ are the beamspace channel matrix and noise vector, respectively.

Using the commonly adopted plane-wave approximation for mmWave frequencies, the channel vectors associated with each UE in the antenna domain can be modeled as [27]

$$\mathbf{h} = \sum_{\ell=0}^{L-1} \alpha_\ell \mathbf{a}(\phi_\ell), \quad \mathbf{a}(\phi) = [e^{j0\phi}, e^{j1\phi}, \ldots, e^{j(B-1)\phi}]^{\mathrm{T}}. \tag{2}$$

Here, L denotes the total number of arriving paths, $\alpha_\ell \in \mathbb{C}$ is the channel gain of the ℓth path, and $\mathbf{a}(\phi_\ell)$ is the complex-valued sinusoid vector, where $\phi_\ell \in [0, 2\pi)$ is determined by the ℓth path's angle-of-arrival. If the number of paths L is small compared to BS antenna array size B, which is the case in mmWave massive MIMO [6], then the beamspace channel vector $\hat{\mathbf{h}} = \mathbf{F}\mathbf{h}$ of each UE will be sparse; this enables low-complexity baseband processing algorithms, such as data detectors and channel estimators [10]–[12], [28].

B. Discrete Fourier Transform (DFT)

The critical ingredient of beamspace algorithms is the spatial DFT. The DFT of a vector $\mathbf{y} \in \mathbb{C}^B$ is given by [29]:

$$[\hat{\mathbf{y}}]_k = \frac{1}{\sqrt{B}} \sum_{n=0}^{B-1} [\mathbf{y}]_n W_B^{kn}, \quad k = 0, 1, \ldots, B-1. \tag{3}$$

Here, $W_B^{kn} = \exp(-j2\pi kn/B)$ are the twiddle factors and $j^2 = -1$. Equivalently, we can write $\hat{\mathbf{y}} = \mathbf{F}\mathbf{y}$, where $\mathbf{F}$ is the unitary DFT matrix, whose (k, n)th entry is $\frac{1}{\sqrt{B}} W_B^{kn}$.

III. SMUL-FFT: Streaming MUltiplierLess FFT

A. Quantizing Twiddle Factors to Sum-of-Powers-of-Two

To avoid the need for hardware multipliers for twiddle factor multiplication, we design an approximate set of twiddle factors whose CSD representation consists of 5 digits with at most 2 nonzero digits [22]. For example, the CSD representation $[1, 0, 0, 0, -1]$ corresponds to $1 \cdot 2^0 + (-1) \cdot 2^{-4}$ and has two nonzero digits. We note that the CSD representation minimizes the number of nonzero digits compared to other number representations [30]. We define Ω as the set of complex numbers within a ring around the unit circle, i.e., $1 - \delta < |z| < 1 + \delta$, $\forall z \in \Omega$ with $\delta = 0.2$, whose real and imaginary components have a 5 digit CSD representation with at most 2 nonzero digits. The first quadrant of such a set is shown in Figure 2.

Let $\mathbf{G}(\cdot) : \mathbb{C}^B \to \mathbb{C}^{B\times B}$ be a function whose output for $\mathbf{u} \in \mathbb{C}^B$, is $\mathbf{G}(\mathbf{u}) = \frac{1}{\sqrt{B}} \left[\mathbf{u}^{\circ 0}, \mathbf{u}^{\circ 1}, \cdots, \mathbf{u}^{\circ (B-1)}\right]$. In particular, for $\mathbf{w} \in \mathbb{C}^B$ with $w_k = W_B^k$, $\mathbf{G}(\mathbf{w}) = \mathbf{F}$. Our aim is to find a vector $\tilde{\mathbf{w}} \in \Omega^B$ that gives the best approximate DFT via $\tilde{\mathbf{F}} = \mathbf{G}(\tilde{\mathbf{w}})$. A straightforward approach would be to map each W_B^k to its closest neighbor in Ω according to $\tilde{w}_k = \arg\min_{z\in\Omega} |W_B^k - z|$. This naïve approach, however, is sub-optimal. Instead, we propose to find a DFT matrix $\tilde{\mathbf{F}}$ that minimizes the normalized mean-square error (NMSE) in the transform of random vectors $\mathbf{x} \in \mathbb{C}^B$ defined as

$$NMSE \triangleq \frac{\mathbb{E}\left[\|\tilde{\mathbf{F}}\mathbf{x} - \mathbf{F}\mathbf{x}\|_2^2\right]}{\mathbb{E}[\|\mathbf{F}\mathbf{x}\|_2^2]} = \frac{\|\tilde{\mathbf{F}} - \mathbf{F}\|_F^2}{\|\mathbf{F}\|_F^2}. \tag{4}$$

Here, the last equality holds for the following two relevant distributions for $\mathbf{x}$: (i) zero-mean complex Gaussian vectors, i.e., $\mathbf{x} \sim \mathcal{CN}(\mathbf{0}, \rho^2\mathbf{I}_B)$, and (ii) complex-valued sinusoids with a random phase, i.e., $\mathbf{x} = \mathbf{a}(\Phi)$ where $\Phi \sim \mathrm{Unif}(0, 2\pi)$.

To design $\tilde{\mathbf{F}}$, we successively quantize each twiddle factor. First, we find the worst four twiddle factors with the largest distance from their nearest neighbors in Ω and quantize them one-by-one. Then, we use a greedy approach to choose which twiddle factor to quantize next based on minimizing the *NMSE*. Suppose we wish to quantize w_b. Then, $\tilde{w}_b$ is the minimizer of the *NMSE* among w_b's four nearest neighbors in Ω, which are the elements that are closest to w_b in Euclidean distance.

Our simulations for $B \in \{32, 64, 128, 256\}$ show that our quantization strategy decreases the *NMSE* by at least 0.9 dB up to 1.5 dB compared to the naïve approach of individually quantizing the twiddle factors to their nearest neighbor in Ω.

B. SMUL-FFT Architecture Details

1) Fully-Unrolled Radix-4 Architecture: For the beamspace transform in (1), the input vectors are generated at baseband sampling rate f_s, which can require billions of transforms per second in mmWave applications. A fully-unrolled pipelined architecture offers the highest possible throughput and lowest latency, while eliminating the need for reorder buffers, twiddle

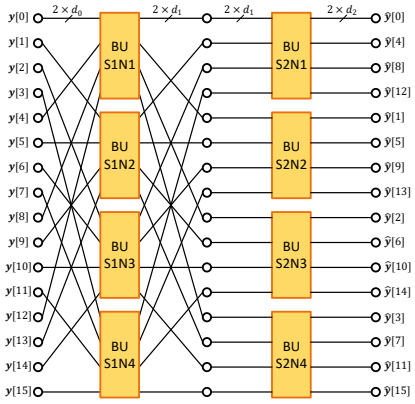


Fig. 3. High-level architecture of a 16-point SMUL-FFT showing the butterfly units (BUs). The yth butterfly in the xth stage is denoted by SxNy.

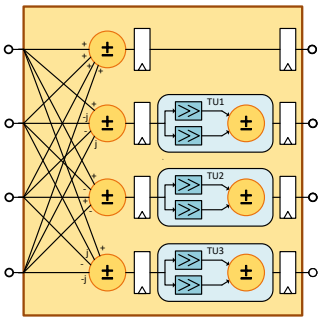
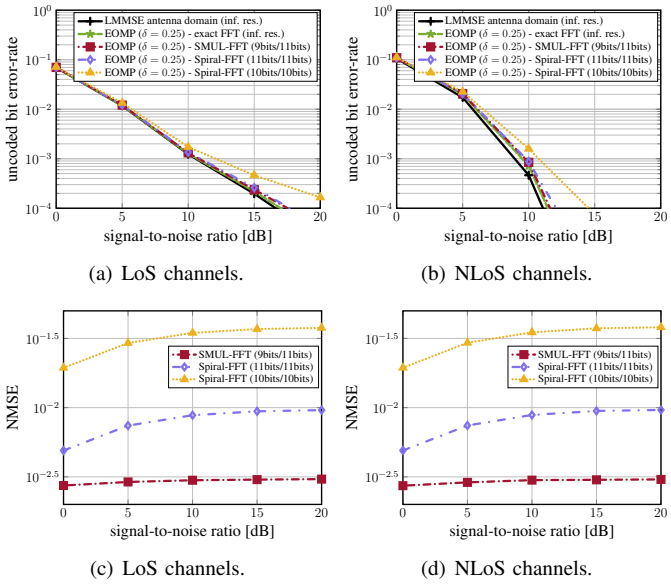


Fig. 4. Internal architecture of a SMUL-FFT radix-4 butterfly.

factor memories, and control circuitry, thereby achieving the highest efficiency in terms of energy per transform and area per throughput—see the end of Section V for more details. An example of the 16-point SMUL-FFT architecture with two stages consisting of radix-4 butterflies is depicted in Figure 3. If B is an even power of two, SMUL-FFT consists of $\log_4(B)$ stages with $B/4$ radix-4 butterflies; if B is an odd power of two, then SMUL-FFT consists of $B/2$ radix-2 butterflies in the first stage and $\lfloor \log_4(B) \rfloor$ stages of $B/4$ radix-4 butterflies.

2) Scaling Schedule and Bitwidth-Growth Profile: Figure 4 depicts the architecture of a radix-4 butterfly, comprising four complex adders/subtractors and three twiddle units (labeled TU). In each 4-input adder, the bitwidth of the adder's output increases by two bits. Therefore, to prevent overflow, the bitwidth of each radix-4 stage must grow by two bits, summing to $\log_2 B$ bits overall. Alternatively, in each stage, we can scale down the results of the additions/subtractions in all butterflies by 2^s, where $s \in \{0, 1, 2\}$, in which case the output of that stage will require $2 - s$ bits more than its inputs, given that s least significant bits (LSBs) are truncated. Note that in the case of $s > 0$, the truncation of s LSBs may degrade accuracy. We used simulations to determine the optimal combination of scaling schedule and bitwidth growth profile that incurs no noticeable performance loss, while minimizing silicon area.

3) Decimation-In-Frequency (DIF): As shown in Figure 3, we use a DIF architecture [29], which simplifies signal routing as we progress through the stages towards the end of the SMUL-FFT. Since the output bitwidth of each stage is larger than or equal to the bitwidth of its inputs, it is beneficial to have simpler routing in stages with larger bitwidth.



(a) LoS channels. (b) NLoS channels.

(c) LoS channels. (d) NLoS channels.

Fig. 5. BER and NMSE simulation results of a mmWave massive MU-MIMO system with $B = 256$ and $U = 16$, for LoS and NLoS channels.

4) Automatic Verilog Generation: In order to facilitate the design of SMUL-FFTs, we developed a MATLAB script that automatically generates Verilog code for the following set of parameters: FFT size, input bitwidth, scaling schedule, and bitwidth growth profile. Since the twiddle factors inside each butterfly are fixed in a fully-unrolled architecture, no memory elements for twiddle factor storage are needed.

General-purpose complex-valued multiplication is typically realized by four real-valued multipliers and two adders. However, since the SMUL-FFT implements each real-valued twiddle factor multiplication with at most one adder (or subtractor) and two shift operations, each complex twiddle unit (TU) inside the radix-4 butterfly (as shown in Figure 4) implements multiplication by the quantized complex-valued twiddle factor with at most six adders/subtractors and arithmetic shift operations. Furthermore, since the twiddle factors are constant and these arithmetic shifts are simply wire selections, avoiding any hardware overhead.

IV. Simulation Results

To evaluate the performance of our SMUL-FFT with the proposed set of twiddle factors, we consider a mmWave massive MU-MIMO system in which $U = 16$ single-antenna UEs transmit 16-QAM symbols to a $B = 256$ antenna ULA BS with $\lambda/2$ antenna spacing. We consider line-of-sight (LoS) and non-LoS (NLoS) channels generated by the QuaDRiGa mmMAGIC UMi model [31] at a 60 GHz carrier frequency. The UEs are randomly placed in a 120° sector within the range of 10 m–110 m, with a minimum angular separation of 1°.

We simulate the uncoded bit error rate (BER) of the EOMP sparsity-exploiting beamspace detector [12] using only $\delta = 1/4$ of all B beams. We perform pilot-based channel estimation followed by BEACHES [4], a beamspace channel vector denoising algorithm. For FFT, we consider (i) SMUL-FFT with 9 bits per real and imaginary component for the inputs and 11 bits for the outputs, (ii) Spiral-FFT with 10-bit inputs

TABLE I
ASIC IMPLEMENTATION RESULTS AND COMPARISON IN TSMC 65 NM CMOS FOR VARIOUS FFT SIZES.

Design	SMUL-FFT				Spiral-FFT			
FFT size B	32	64	128	256	32	64	128	256
Input/output bitwidth[a]	9/10	9/11	9/11	9/11	10/10	11/11	11/11	11/11
Clock frequency [MHz]	750	500	500	500	938	938	833	833
Core area [mm^2]	0.22	0.32	0.91	2.03	0.34	0.72	2.06	4.08
Power consumption [W][b]	0.14	0.21	0.57	1.35	0.3	0.73	1.92	4.02
Latency [clock cycles]	18	18	20	20	17	16	25	22
Area-efficiency [mm^2/(G transforms/s)]	0.29 (80%)	0.64 (84%)	1.82 (73%)	4.06 (83%)	0.36	0.76	2.47	4.89
Energy-efficiency [nJ/transform]	0.18 (56%)	0.41 (53%)	1.14 (49%)	2.7 (56%)	0.32	0.77	2.3	4.82

[a] Bitwidth per real and imaginary part to achieve similar performance as a floating-point FFT (see Section IV).
[b] Extracted with stimuli from postlayout simulation in the typical-typical process corner at 25°C and nominal voltage of 1.2 V.

and outputs, and (iii) Spiral-FFT with 11-bit inputs and outputs. In the Spiral-FFT code generator [26], the input and output bitwidths are equal and the outputs of each radix-4 stage are either scaled down by 4 and truncated or left unscaled, which can result in overflow. We used the scaled version for Spiral, as the unscaled version incurred a significant performance loss, due to the overflow. Since no bit-true software model for the Spiral-FFT is available, we used HDL simulation to obtain the output corresponding to each input vector.

Figures 5(a) and (b), show the BER versus signal-to-noise ratio (SNR) for LoS and NLoS channels, respectively, where we include the conventional antenna-domain linear minimum mean square (LMMSE) detector as a baseline. The number of integer and fraction bits for the inputs of each FFT design are optimized via simulations. We observe in Figure 5, that the SMUL-FFT with 9-bit inputs and 11-bit outputs closes the performance gap to a floating-point FFT, while the Spiral-FFT requires 11-bit inputs and outputs to achieve similar performance. The main reason is the conservative scaling schedule of the Spiral-FFT which loses performance by scaling down results of each stage and truncating the LSBs. In Figure 5 (c) and (d), we show the NMSE (defined in (4)) for the SMUL-FFT and Spiral-FFT, with the same parameters as in Figure 5 (a) and (b). We see that the SMUL-FFT with 9-bit inputs and 11-bit outputs, achieves lower NMSE than the Spiral-FFT with 11-bit inputs and outputs for LoS and NLoS channels.

V. ASIC IMPLEMENTATION RESULTS

We now provide ASIC implementation results for the proposed SMUL-FFT and compare them to Spiral-FFT designs [26], which we chose as our baseline for the following reasons. First, Spiral-FFTs achieve state-of-the-art hardware-efficiency, on par with that of Xilinx FFT IPs [32], [33]. Second, as discussed at the end of this section, fully-unrolled FFT architectures achieve the highest energy-efficiency and maximum throughput, making them particularly suitable for mmWave massive MIMO systems operating at very high sampling rates. To the best of our knowledge, only Spiral-FFT enables us to generate fully-unrolled architectures. Third, to enable a fair comparison, we need to optimize the key architecture parameters (signal bitwidths, scaling schedule, radix, etc.) for each design—this is not practicable for the majority of existing, custom-designed FFT implementations.

Table I provides the implementation results for SMUL-FFT and Spiral-FFT designs for $B \in \{32, 64, 128, 256\}$ in 65 nm TSMC. For each design, we performed an area-delay sweep to identify the clock frequency that minimizes the area-delay product, which we report in Table I as the clock frequency. For each design, we use the smallest input/output bitwidth pair that results in no noticeable performance loss compared to a floating-point FFT (cf. Section IV). All area and power consumption results in Table I are extracted after the layout stage. For the power consumption, we used node switching activity extracted from postlayout simulation with realistic stimuli generated using NLoS channels and the simulation setup described in Section IV. Since the SMUL-FFT and Spiral-FFT are streaming fully-unrolled architectures, their throughput is equal to f_{clk} million vectors per second, where f_{clk} is the clock frequency in MHz.[2] Since Spiral-FFTs achieve a higher clock frequency than our designs, to enable a fair comparison in Table I, we provide area-efficiency in terms of area divided by throughput and energy-efficiency in terms of the energy per transform. To facilitate this comparison, we provide percentage values in parentheses for SMUL-FFT, which are obtained by dividing the energy- and area-efficiency numbers of SMUL-FFT columns, by the corresponding numbers from the Spiral-FFT columns.

From Table I, we see that the SMUL-FFT requires 17% to 27% lower area for the same throughput compared to the Spiral-FFT, and offers power savings ranging from 44% to 51%. The area and power savings of SMUL-FFT come in part from the fact that twiddle factor multiplications in SMUL-FFT are realized by adders and shifters instead of dedicated multipliers, and in part from better scaling schedule which allows for smaller input bitwidth to achieve the same (and often better) performance compared to Spiral-FFT.

In order to demonstrate that the fully-unrolled architecture achieves superior energy-efficiency, we also implemented parallel pipelined versions of the Spiral-FFT with streaming width of four complex samples per clock cycle. A 256-point Spiral-FFT with streaming width of four, achieves a clock frequency of $625\,\text{MHz}$, occupies $0.78\,\text{mm}^2$, and consumes $250\,\text{mW}$. Since $256/4 = 64$ clock cycles are required to complete a 256-point FFT with this architecture, the energy consumed per transform is $25.6\,\text{nJ}$, which is $5\times$ higher than that of the fully-unrolled Spiral-FFT—this confirms that fully-unrolled pipelined architectures offer the highest efficiency.

[2] We note that the implementations in Table I achieve relatively high clock frequencies for a 65 nm CMOS process, which is a result of fine-grained pipelining and relatively low bitwidths (i.e., 9 bits to 11 bits). The higher clock frequency of the Spiral-FFT designs is due to more aggressive pipelining.

VI. Conclusions

Beamspace processing in all-digital mmWave massive MU-MIMO wireless systems requires high-throughput DFT engines to transform B-dimensional vectors at the baseband sampling rate. For such applications, we have proposed an efficient, fully-unrolled Streaming MUltiplierLess (SMUL) FFT, by designing an approximate set of twiddle factors whose CSD representation has 5 digits with no more than 2 nonzero digits. This strategy enables multiplication by real/imaginary parts of the quantized twiddle factors to be realized with at most one adder/subtractor and trivial shift operations. Our simulation results demonstrate that the proposed SMUL-FFT achieves near floating point error-rate performance in mmWave massive MU-MIMO systems. Furthermore, ASIC implementation results in 65 nm CMOS demonstrate that the proposed SMUL-FFT consumes nearly 50% less energy per transform compared to the state-of-the-art Spiral-FTT designs [26]. Therefore, the proposed SMUL-FFT greatly mitigates the DFT implementation bottleneck in mmWave beamspace processing.

There exist many avenues for future work. Concretely, the design of beamspace transforms for two-dimensional or non-uniform antenna arrays, distributed or cell-free massive MIMO systems, hybrid analog-digital architectures, and downlink processing are interesting open research problems.

References

[1] T. S. Rappaport, S. Sun, R. Mayzus, H. Zhao, Y. Azar, K. Wang, G. N. Wong, J. K. Schulz, M. Samimi, and F. Gutierrez, "Millimeter wave mobile communications for 5G cellular: It will work!" *IEEE Access*, vol. 1, pp. 335–349, May 2013.

[2] T. S. Rappaport, R. W. Heath Jr., R. C. Daniels, and J. N. Murdock, *Millimeter Wave Wireless Communications*. Prentice Hall, 2015.

[3] E. G. Larsson, F. Tufvesson, O. Edfors, and T. L. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 186–195, Feb. 2014.

[4] S. H. Mirfarshbafan, A. Gallyas-Sanhueza, R. Ghods, and C. Studer, "Beamspace channel estimation for massive MIMO mmWave systems: Algorithm and VLSI design," *IEEE Trans. Circuits Syst. I*, pp. 1–14, Sep. 2020.

[5] A. Alkhateeb, O. El Ayach, G. Leus, and R. W. Heath Jr., "Channel estimation and hybrid precoding for millimeter wave cellular systems," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 831–846, Oct. 2014.

[6] P. Schniter and A. Sayeed, "Channel estimation and precoder design for millimeter-wave communications: The sparse way," in *Proc. Asilomar Conf. Signals, Syst., Comput.*, Nov. 2014, pp. 273–277.

[7] J. Deng, O. Tirkkonen, and C. Studer, "mmWave channel estimation via atomic norm minimization for multi-user hybrid precoding," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2018, pp. 1–6.

[8] A. Alkhateeb, G. Leus, and R. W. Heath Jr., "Limited feedback hybrid precoding for multi-user millimeter wave systems," *IEEE Trans. Wireless Commun.*, vol. 14, no. 11, pp. 6481–6494, Nov. 2015.

[9] J. Lee, G. Gil, and Y. H. Lee, "Channel estimation via orthogonal matching pursuit for hybrid MIMO systems in millimeter wave communications," *IEEE Trans. Commun.*, vol. 64, no. 6, pp. 2370–2386, Jun. 2016.

[10] M. Abdelghany, U. Madhow, and A. Tölli, "Beamspace local LMMSE: An efficient digital backend for mmWave massive MIMO," in *Proc. IEEE Int. Workshop Signal Process. Advances Wireless Commun. (SPAWC)*, Aug. 2019, pp. 1–5.

[11] M. Mahdavi, O. Edfors, V. Öwall, and L. Liu, "Angular-domain massive MIMO detection: Algorithm, implementation, and design tradeoffs," *IEEE Trans. Circuits Syst. I*, vol. 67, no. 6, pp. 1948–1961, Jan. 2020.

[12] S. H. Mirfarshbafan and C. Studer, "Sparse beamspace equalization for massive MU-MIMO mmWave systems," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, May 2020, pp. 1773–1777.

[13] 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.

[14] S. Bhattacharyya, E. Deprettere, R. Leupers, and J. Takala, *Handbook of Signal Processing Systems: Third Edition*, 2013.

[15] X. Shih, Y. Liu, and H. Chou, "48-mode reconfigurable design of SDF FFT hardware architecture using radix-32 and radix-23 design approaches," *IEEE Trans. Circuits Syst. I*, vol. 64, no. 6, pp. 1456–1467, May 2017.

[16] N. Le Ba and T. T. Kim, "An area efficient 1024-point low power radix-22 FFT processor with feed-forward multiple delay commutators," *IEEE Trans. Circuits Syst. I*, vol. 65, no. 10, pp. 3291–3299, May 2018.

[17] S. Chen, S. Huang, M. Garrido, and S. Jou, "Continuous-flow parallel bit-reversal circuit for MDF and MDC FFT architectures," *IEEE Trans. Circuits Syst. I*, vol. 61, no. 10, pp. 2869–2877, Jul. 2014.

[18] M. Ayinala, M. Brown, and K. K. Parhi, "Pipelined parallel FFT architectures via folding transformation," *IEEE Trans. VLSI Syst.*, vol. 20, no. 6, pp. 1068–1081, May 2011.

[19] J. Brady, N. Behdad, and A. M. Sayeed, "Beamspace MIMO for millimeter-wave communications: System architecture, modeling, analysis, and measurements," *IEEE Trans. Antennas Propag.*, vol. 61, no. 7, pp. 3814–3827, Jul. 2013.

[20] P. Skrimponis, S. Dutta, M. Mezzavilla, S. Rangan, S. H. Mirfarshbafan, C. Studer, J. Buckwalter, and M. Rodwell, "Power consumption analysis for mobile mmwave and sub-THz receivers," in *2020 IEEE 6G Wireless Summit*, Mar. 2020, pp. 1–5.

[21] Z. M. Enciso, S. Hadi Mirfarshbafan, O. Castañeda, C. J. Schaefer, C. Studer, and S. Joshi, "Analog vs. digital spatial transforms: A throughput, power, and area comparison," in *Proc. IEEE Int. Midwest Symp. Circuits and Syst. (MWSCAS)*, Sep. 2020, pp. 125–128.

[22] W. Perera, "Architectures for multiplierless fast Fourier transform hardware implementation in VLSI," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 35, no. 12, pp. 1750–1760, Dec. 1987.

[23] S. C. Chan and P. M. Yiu, "An efficient multiplierless approximation of the fast Fourier transform using sum-of-powers-of-two (SOPOT) coefficients," *IEEE Signal Processing Letters*, vol. 9, no. 10, pp. 322–325, Dec. 2002.

[24] Wei Han, T. Arslan, A. T. Erdogan, and M. Hasan, "Multiplier-less based parallel-pipelined FFT architectures for wireless communication applications," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 5, May 2005, pp. v/45–v/48 Vol. 5.

[25] M. Garrido, R. Andersson, F. Qureshi, and O. Gustafsson, "Multiplierless unity-gain SDF FFTs," *IEEE Trans. VLSI Syst.*, vol. 24, no. 9, pp. 3003–3007, Apr. 2016.

[26] P. Milder, F. Franchetti, J. C. Hoe, and M. Püschel, "Computer generation of hardware for linear digital signal processing transforms," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 17, no. 2, Apr. 2012. [Online]. Available: https://doi.org/10.1145/2159542.2159547

[27] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge Univ. Press, 2005.

[28] A. Gallyas-Sanhueza, S. H. Mirfarshbafan, R. Ghods, and C. Studer, "Sparsity-adaptive beamspace channel estimation for 1-bit mmWave massive MIMO systems," in *Proc. IEEE Int. Workshop Signal Process. Advances Wireless Commun. (SPAWC)*, Aug. 2020, pp. 1–5.

[29] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*. Prentice Hall, 1998.

[30] In-Cheol Park and Hyeong-Ju Kang, "Digital filter synthesis based on an algorithm to generate all minimal signed digit representations," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 21, no. 12, pp. 1525–1529, Dec. 2002.

[31] S. Jaeckel, L. Raschkowski, K. Börner, L. Thiele, F. Burkhardt, and E. Eberlein, "QuaDRiGa - quasi deterministic radio channel generator user manual and documentation," Fraunhofer Heinrich Hertz Institute, Tech. Rep. v2.0.0, Aug. 2017.

[32] G. Nordin, P. A. Milder, J. C. Hoe, and M. Püschel, "Automatic generation of customized discrete Fourier transform IPs," in *Proc. IEEE Design Automation Conf. (DAC)*, Sep. 2005, pp. 471–474.

[33] P. A. Milder, F. Franchetti, J. C. Hoe, and M. Puschel, "Formal datapath representation and manipulation for implementing DSP transforms," in *Proc. IEEE Design Automation Conf. (DAC)*, Jul. 2008, pp. 385–390.