A Reconfigurable FFT Architecture for Variable-Length and Multi-Streaming OFDM Standards

Padma Prasad Boopal, Mario Garrido Gálvez and Oscar Gustafsson

Conference Publication

N.B.: When citing this work, cite the original article.

A Reconfigurable FFT Architecture for Variable-Length and Multi-Streaming OFDM Standards

Padma Prasad Boopal, Mario Garrido, and Oscar Gustafsson
Department of Electrical Engineering, Linköping University
SE-581 83 Linköping, Sweden
E-mail: `padbo235@student.liu.se, mariog@isy.liu.se, oscarg@isy.liu.se`

Abstract—This paper presents a reconfigurable FFT architecture for variable-length and multi-streaming WiMax wireless standard. The architecture processes 1 stream of 2048-point FFT, up to 2 streams of 1024-point FFT or up to 4 streams of 512-point FFT. The architecture consists of a modified radix-2 single delay feedback (SDF) FFT. The sampling frequency of the system is varied in accordance with the FFT length. The latch-free clock gating technique is used to reduce power consumption.

The proposed architecture has been synthesized for the Virtex-6 XCVLX760 FPGA. Experimental results show that the architecture achieves the throughput that is required by the WiMax standard and the design has additional features compared to the previous approaches. The design uses 1% of the total available FPGA resources and maximum clock frequency of 313.67 MHz is achieved. Furthermore, this architecture can be expanded to suit other wireless standards.

Index Terms—Fast Fourier Transform (FFT), Radix-2, Single Delay Feedback (SDF), Decimation in Time (DIT), Variable-length, Multi-streaming, Pipelined Architecture, Worldwide Interoperability for Microwave Access (WiMax), OFDM.

I. INTRODUCTION

In wireless communication systems, a key factor influencing performance is the system throughput. In recent days, wireless systems have been rapidly developed and key research is done to increase the transmission rate of the system. WiMax 802.16e/m is a wireless standard that combines the MIMO technology and the OFDM technology to achieve high throughput rates, thereby achieving higher data transmission rates. In a WiMax MIMO-OFDM system, the FFT is the key signal processing algorithm. Multiple spatial streams are used in MIMO systems to increase the system performance. Transmission of multiple spatial streams requires FFT processing of multiple data streams. In general, multiple FFT processors are added to the wireless system to handle multiple data streams. In comparison to the use of a single FFT processor, the usage of multiple processors demands more hardware resources and increases the power consumption. In addition to this, WiMax wireless standards transmit data in different defined channel bandwidths (5 MHz, 10 MHz or 20 MHz). This demands scaling the FFT (variable FFT length) to the defined channel bandwidth to maintain a constant carrier spacing.

Therefore, in order to meet the requirements of the standard, a reconfigurable architecture that can support variable length and multiple streams is needed. This has led to significant research on pipelined FFT architectures [1]–[10]. As a result, several variable-length and multi-streaming architectures have been proposed in literature. A variable-length architecture for WiMax systems that can process FFTs of 512, 1024, 2048 or 4096 points is presented in [3]. For WLAN standards, a 64-point multi-streaming architecture is proposed in [5]. A dynamic voltage and frequency scaling pipelined architecture is proposed in [6], which can process up to 8 streams in parallel. A multi-standard design for WPAN/WLAN/WMAN is proposed in [7]. Variable length for 128/256 points and parallel processing of up to 4 stream is presented in [8]. Finally, the architecture in [9] processes 8 samples of data in parallel for high-throughput WPAN systems. From previous works it can be observed that most of the architectures in the literature only support either variable length or multi-streaming, but not both of them simultaneously.

This paper presents a reconfigurable FFT architecture for WiMax. The architecture supports both variable length and multi-streaming simultaneously. It consists of a modified single delay feedback (SDF) radix-2 FFT architecture. In order to reduce the power consumption, the latch-free clock gating technique is used to clock the modules only when they are needed. Unused stages of the pipeline are powered down. For multi-stream processing, the coefficient storage is organized in order to reduce the number of memory accesses.

The paper is organized as follows. Section II discusses the radix-2 algorithm. Section III presents the new variable-length multi-streaming FFT architecture. Section IV discusses considerations about the hardware implementation and Section V present the strategy that has been followed to be able to combine variable length and multi-streaming in the same design. Section VI shows the experimental results. Section VII presents the power consumption results and discusses the savings that are achieved in the different configurations. Finally, Section VIII concludes the paper.

II. THE RADIX-2 FFT ALGORITHM

The N-point DFT of an input data sequence is given by

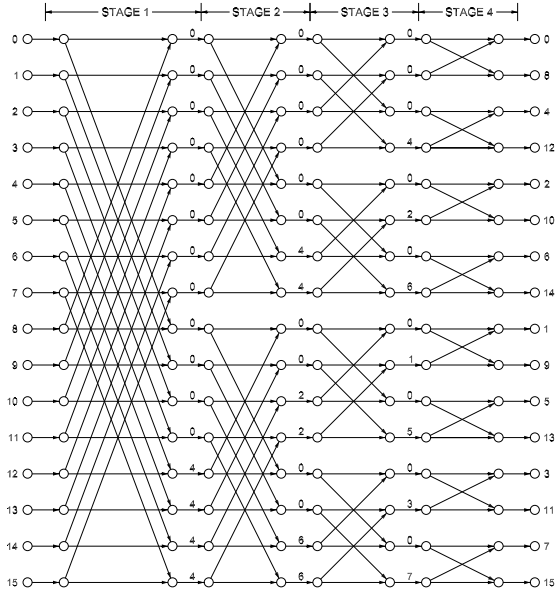$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}nk} \tag{1}$$

Fig. 1. Flow graph of a 16-point radix-2 DIT FFT.

Fig. 2. Proposed variable-length multi-streaming FFT architecture.

Fig. 3. FFT input schemes for the proposed architecture.

where N represents the number of points of the DFT, $x[n]$ are the samples in the time domain and $X[k]$ represents the signal in the frequency domain. The terms $e^{-j\frac{2\pi}{N}nk}$ are called twiddle factor and are usually represented by W_N^{nk}. The twiddle factors are complex numbers of magnitude one, so they define rotations in the complex plane.

The Cooley-Tukey FFT algorithm is the most widely used method and an efficient way of computing the DFT. While a DFT computation of length N takes $O(N^2)$ arithmetic operations, an FFT for the same length requires only $O(N \log N)$ operations. Radix-2 decimation in frequency (DIF) and radix-2 decimation in time (DIT) are the simplest algorithms in terms of design complexity. The basic operation of a radix-2 algorithm is called butterfly due to the shape of the data flow diagram. The butterfly operation comprises of one addition, one subtraction and one multiplication operations.

Figure 1 shows the signal flow graph of a 16-point radix-2 DIT FFT. For radix-2, the number of stages for an N-point FFT is given by $n = \log_2 N$ and the butterflies at each stage calculate 2-point FFTs. The signals at the input represent the time domain sequence, $x[n]$, and the signals at the output represent the frequency domain sequence, $X[k]$. The numbers between the stages represent the rotations by the twiddle factors. It can be noted from the graph that the inputs are in natural order, whereas the outputs are in a different order. The output sequence is in bit reversal [11] with respect to the input sequence. The bit reversal of a binary number $b_0, b_1, b_2...b_{n-1}$ is given by $b_{n-1}, ..., b_2, b_1, b_0$. Note that, contrary to the general belief, a DIT FFT can be expressed with inputs in natural order and outputs in bit reversal, as shown in Figure 1. This was demonstrated in [12].

III. PROPOSED ARCHITECTURE

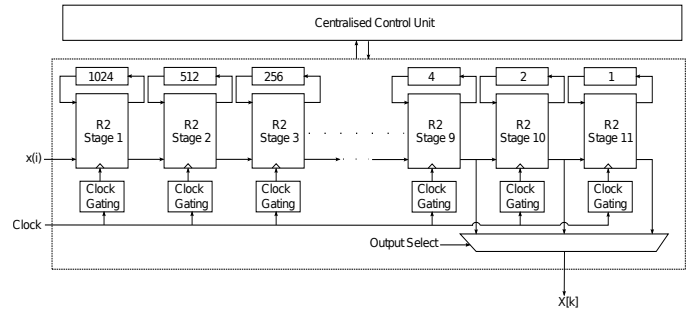Figure 2 shows the proposed variable-length and multi-stream FFT architecture. The architecture is a modified single delay feedback (SDF) pipelined architecture. Each stage calculates a radix-2 butterfly and the FFT is computed using the decimation in time (DIT) algorithm. The DIT algorithm has the property that the first s stages of the algorithm always calculate an FFT of $N = 2^s$ points, for any s [13]. Accordingly, the DIT algorithm has been chosen so that the initial stages of the pipelined architecture can be shared between schemes with different FFT lengths and multiple streams. The design expands into a multi-stream processing architecture by interleaving the input streams and changing the system sampling frequency.
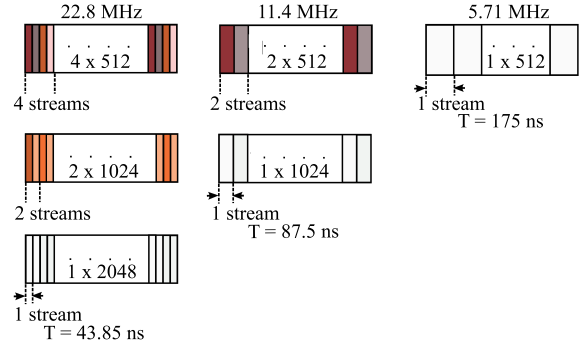
The number of stages required to calculate a 2048-point FFT is $n = \log_2(2048)$. Hence, the architecture consists of $n = 11$ stages. The FFT schemes for the input $x[i]$ to the architecture are shown in Figure 3. The architecture can process either a 2048-point FFT, or two interleaved streams of 1024-point FFTs, or four interleaved streams of 512-point FFTs at 22.8 MHz. The system can also be configured to process one stream of 1024 points or two interleaved streams of 512 points with a clock of 11.4 MHz. Finally, a single stream of 512 points can be calculated at 5.71 MHz. When the size of the FFT is smaller than 2048, some of the last stages of the architecture are not used. As nine stages are required to calculate a 512-point FFT, the outputs for this case are selected from the $9th$ stage by a multiplexer, as shown in Fig. 2. For this configuration, stages 10 and 11 are powered down. Likewise, ten stages are required to calculate a 1024-point FFT and the output is selected from the $10th$ stage, while stage 11 is powered down.
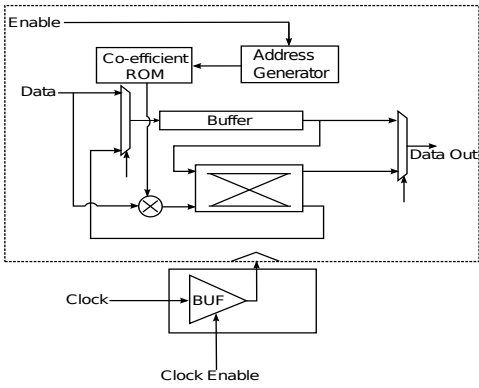
Fig. 4.   A modified SDF stage of the proposed architecture.

Fig. 5.   Structure of the buffer.

IV. HARDWARE IMPLEMENTATION

Figure 4 shows a single stage of the proposed pipelined architecture. Each pipelined stage consists of a radix-2 butterfly element, a complex multiplier, a coefficient memory, an address generator for the coefficient memory and data management units. This results in a total of eleven butterfly elements and ten complex multipliers for the entire design. Additionally, the design consists of a centralized control unit to synchronize the data flow.

The butterfly unit in the proposed design consists of a complex adder and a complex subtractor. The butterfly unit produces the complex sum and complex difference of the real and imaginary parts of the inputs in accordance with

$$
\begin{aligned}
X_\mathrm{r} &= x_\mathrm{r} + y_\mathrm{r} \\
X_\mathrm{i} &= x_\mathrm{i} + y_\mathrm{i} \\
Y_\mathrm{r} &= x_\mathrm{r} - y_\mathrm{r} \\
Y_\mathrm{i} &= x_\mathrm{i} - y_\mathrm{i}
\end{aligned}
\tag{2}
$$

where $x_\mathrm{r} + jx_\mathrm{i}$ and $y_\mathrm{r} + jy_\mathrm{i}$ are the inputs to the butterfly and $X_\mathrm{r} + jX_\mathrm{i}$ and $Y_\mathrm{r} + jY_\mathrm{i}$ are the outputs.

The functional implementation of the complex multiplier can be explained by

$$
\begin{aligned}
y_\mathrm{r} &= a_\mathrm{r}C - a_\mathrm{i}S = a_\mathrm{r}(C + S) - (a_\mathrm{r} + a_\mathrm{i})S \\
y_\mathrm{i} &= a_\mathrm{r}S + a_\mathrm{i}C = a_\mathrm{r}(C + S) + (a_\mathrm{i} - a_\mathrm{r})C
\end{aligned}
\tag{3}
$$

where $a_\mathrm{r} + ja_\mathrm{i}$ is the input of the complex multiplier, $C + jS$ is the rotation coefficient and $y_\mathrm{r} + jy_\mathrm{i}$ is the output. In general, the direct implementation of a complex multiplier requires four real multipliers [14]. However, by exploiting the equations of a complex multiplication, the number of multipliers can be reduced to three [15], as shown in equation (3). The proposed architecture uses this approach with three multipliers. Note also that in Fig. 4 the rotator is connected to the lower input of the butterfly, due to the use of the DIT decomposition. For this decomposition, any rotation is only connected to the lower input of a butterfly, as can be observed in Fig. 1. Accordingly, the output of the rotator in equation (3), $y_\mathrm{r} + jy_\mathrm{i}$, correspond to the input $y_\mathrm{r} + jy_\mathrm{i}$ in equation (2).

The buffer structure used in this design is shown in Figure 5. The basic component of the buffer is a shift register. The input data are written to the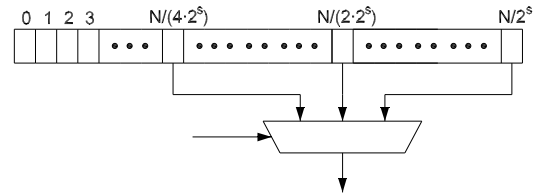 first register and shifted to consecutive registers. Two separate shift registers are used to store the real values and imaginary values. As the depth of the buffers are different for different stages of the pipelined design, the inputs of the buffer are parameterizable to use the same design in all the pipelined stages. The buffer outputs can be selected at one of the three locations $N/2^s$, $N/(2 * 2^s)$ or $N/(4 * 2^s)$. This allows for supporting FFTs of variable length, as well as multi-streaming. In Fig. 5, N represents the larger FFT that the architecture calculates. For the particular case of 2048, 1024 and 512 points, $N = 2048$ and, therefore, the total length of the buffers at any stage, s, is calculated as $L = N/2^s = 2048/2^s$. This corresponds to the buffer sizes in Fig. 2.

To store the twiddle factors, read only memories (ROMs) are used. The size of the ROM in each stage, s, is 2^{s-1}. In each stage, two synchronous single port memory ROMs are used to store the real and imaginary values separately. The twiddle factor word length is quantized to match the input word length. To address the ROM, a simple up-counter is used.

Complex multiplexers are used as switching circuits. The multiplexers pass one of the complex inputs to the output based on the select signal, which is driven by the control unit.

A centralized control unit is used to synchronize and control the data flow path. The control unit consists of an eleven-bit up-counter and few other logical control signals. When the FFT input data arrive, the counter is enabled. The multiplexers in each stage switch the input every $N/2^s$ clock cycles. Hence, each bit of the counter can be directly connected to the select pin of the multiplexer in each stage.

V. VARIABLE LENGTH AND MULTI-STREAMING

To illustrate that the proposed architecture can process both variable length and multi-streaming, Figure 6 presents a signal flow graph for a scheme with one 16-point DIT FFT and two streams of 8-point DIT FFT. This example is chosen for the simplicity of the explanation, and can be generalized to larger FFTs. In the signal flow graph, $x[n]_{16}$ represents the 16-point input data sequence and $X[k]_{16}$ represents the corresponding FFT output samples. Likewise, $x[n]_8$ represents the two streams of 8-point input data sequence and $X[k]_8$ is the FFT output of the two streams. Note that the calculation of the 8-point FFT only requires three stages, so the outputs are taken after the third stage. For the 16-point, the complete signal flow graph is obtained by joining stage 3 with stage 4, and the outputs are taken after stage 4.

In case of two 8-point FFTs, the inputs indexed as $0, 1, 2, 3, 4, 5, 6, 7$ are the data samples of the first 8-point FFT stream and the inputs indexed as $0', 1', 2', 3', 4', 5', 6', 7'$ are
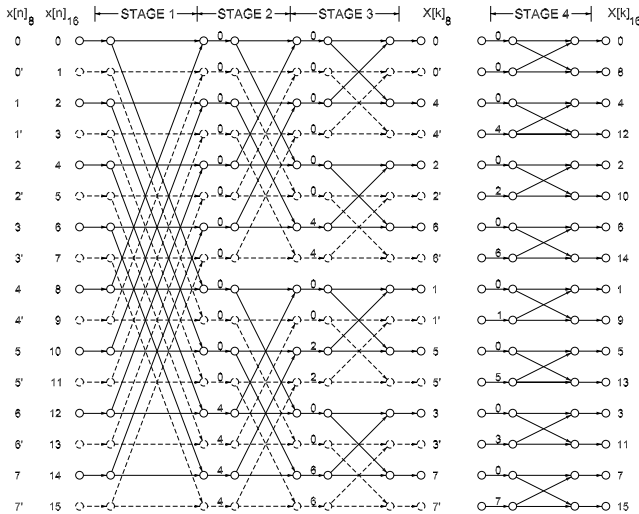
Fig. 6. Flow graph of the variable-length DIT FFT.

Hardware Resource	Utilization	Total
Number of slice registers (Flip Flops)	6,650 (1%)	948,480
Number of slice LUTs	6,614 (1%)	474,240
Number used as Memory	4,392 (3%)	132,480
Number of occupied slices	3,178 (2%)	118,560
Number of DSP48 elements	27 (3%)	864
Number of bonded IOBs	67 (5%)	1,200

the data samples of the second stream. The input streams are interleaved and given as input to the system. In Fig. 6, the solid and dash lines represent the FFT flow graph for the first and second 8-point streams, respectively. As mentioned before, the output $X[k]_8$ is taken from the third stage. The outputs of both 8-point streams are also interleaved. It can be observed that the twiddle factors are the same in any stage, irrespective of the FFT length or the number of streams.

The timing of the data flow can be explained as follows. In stage 1, the first eight samples of the input sequence are buffered. On arrival of the ninth sample, the butterfly operation is enabled. It calculates the complex sum and complex difference of the first sample (stored in the buffer) and the ninth sample. One of the outputs of the butterfly is propagated to the next stage through the multiplexer. The second output of the butterfly is fed back to the buffer and stored there. As the DIT algorithm is used, trivial rotations are calculated at the input of the second stage. The above steps are repeated for the rest of the samples in the input stream.

To compute the FFT of two 8-point streams, the input data samples of the streams are first interleaved and then fed to the input stage. The buffer stores the first four samples of the first stream and the first four samples of the second stream. The order of the samples stored in the buffer is $0, 0', 1, 1', 2, 2', 3, 3'$. On arrival of the ninth sample, the butterfly is enabled to perform the sum and difference operations of the samples 0 and 4. The output of the butterfly that does not have to be rotated is passed on to the next stage by the multiplexer with the appropriate *select* signal from the control unit. The second output of the butterfly is fed back to the buffer.

In the next stages, the data management of the radix-2 structure is analogous to that in the first stage. Likewise, the operations explained for calculating one stream of 16-point FFT and two streams of 8-point FFT can be expanded to calculate multiple streams of higher order FFTs with more stages of the proposed pipelined architecture. Finally, when calculating an FFT of a length smaller than the maximum

length supported by the architecture, the last stages of the pipelined architecture can be powered down. Section VII analyzes the power consumption that is saved by powering down unused stages in the proposed FFT.

VI. EXPERIMENTAL RESULT

To analyze the performance of the proposed FFT architecture, the FFT was implemented in VHDL and synthesized using the Xilinx Integrated Simulation Environment (ISE) tool, Version 12.4i. The target device was a Virtex-6 XCVLX760 FPGA, which is specially designed for low power applications. Table I summarizes the device utilization of the proposed architecture after synthesis. The FFT architecture occupies 3,178 slices, i.e, a 2% of the 118,560 slices that are available in the FPGA. Out of 948,480 flip flops, the architecture uses 6,650. It also use 1% of the 474,240 available look up tables (LUTs). The complex rotations of the FFT are synthesized to fit in the DSP slices of the FPGA. The number of DSP48E elements used is 27, a 3% of the 864 available DSP48Es.

The results of timing analysis using the timing analyzer in the Xilinx ISE tool indicate a maximum clock frequency of 313.67 MHz.

VII. POWER CONSUMPTION

The power estimation of the implemented architecture was done using the Xilinx power analyzer tool. Xilinx power analyzer is an integrated tool of the Xilinx ISE that does power analysis on post-implemented place and route designs. The tool takes information of the clock nets and its corresponding frequencies entered in a power constraints file. The tool imports simulation data from ModelSim in VCD and SAIF file formats. These files help in a quick set up time and provide information about toggling nets in the design. Other environmental settings such as temperature are set to the chip production values. The tool allows to set clock frequencies for individual design elements such as clocks, logic, I/Os, BRAMs, and DSP elements of the implemented design.

Figure 7 presents the power consumption details for different configurations. The peak power consumption is attributed to the FFT length and the clock frequency. For a 2048-point FFT the clock frequency is 22.8 MHz and the peak power consumption is 64.46 mW. As the number of input samples is larger in this case compared to other configurations, the amount of resources consumed is also bigger. When powering down the unused modules using clock gating, the power consumption is reduced to 60.98 mW, saving 5.39%.
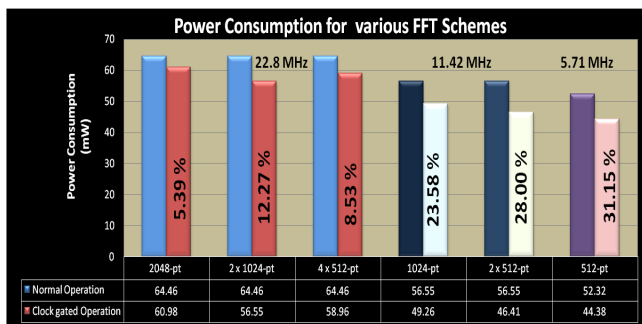
Fig. 7. Power consumption for various FFT schemes.

TABLE II
COMPARISON OF VARIABLE-LENGTH AND MULTI-STREAMING FFTs

PARAMETERS	Proposed	[7]	[8]
Technology	FPGA	ASIC	ASIC
FFT size (N)	512/1024/2048	256/512/1024	64/128
Multi-Streaming Approach	Time Multiplexed	Parallel Streams	Parallel Streams
Number of Streams	Up to 4	Up to 4	Up to 4
Complex Adders	$4\log_4 N$	$32\log_4 N$	$32\log_4 N$
Non-trivial Rotators	$2\log_4 N$	$8\log_4 N$	$8\log_4 N$
Memory Size	N	N	$4N$
Max. Clk (MHz)	313.67	300	-
Freq. Scalability	Yes	No	No
Power Scalability	Yes	Yes	No
Power (mW)	66.48 @ 22.8 MHz	507 @ 300 MHz	-

For a single-stream 1024-point FFT, the power consumption is 56.55 mW. This reduction is due to the use of less hardware resources due to the smaller FFT size, and the half of the clock frequency to 11.42 MHz. Furthermore, as the last stages is not used, it can be powered down, leading to savings of 23.58%.

Similarly, for a single-stream 512-point FFT, the power consumption is 52.32 mW, due to a lower resource utilization and a smaller clock frequency of 5.71 MHz. On powering down unused stages and using clock gating, the power consumption is reduced to 44.38 mW, i.e., power savings of 31.15%.

For two interleaved streams of 1024 points normal and powered down operation consume 56.55 and 49.26 mW, respectively. For four interleaved streams of 512 points, normal mode consumes 64.46 mW and power down takes 58.98 mW.

Table II compares FFT architectures that support variable-length and multi-streaming simultaneously. As these approaches target different OFDM standards with different demands on performance and FFT sizes, a direct comparison of hardware resources and throughput is not appropriate. However, the approaches can be compared qualitatively. In this sense, the main difference of the proposed approach with respect to previous ones is that it implements multi-streaming by time multiplexing the streams in a single flow, whereas previous approaches process several samples in parallel. This makes our design more compact and reduces the number of adders, rotators and memory. Furthermore, the proposed approach presents scalability in frequency and power by using

reconfigurability. This serves to adapt the design to any WiMax configuration and minimize the power consumption.

VIII. CONCLUSION

This papers presents a new pipelined FFT hardware architecture that supports variable-length and multi-streaming simultaneously, based on a time multiplexing of the streams. This combination of features allows for many different configurations. Specifically, the design targets the WiMax wireless OFDM standard. Contrary to previous works, the proposed FFT covers all WiMax configurations, from a single-stream 2048-point FFT to four simultaneous streams of 512 points.

Furthermore, reconfigurability techniques are used to minimize the power consumption. On the one hand, the design provides frequency scalability. Thus, the clock frequency can be varied depending on the configuration that is needed, instead of using the highest frequency for all the cases. On the other hand, some stages of the FFT architecture are not needed for those configurations with smaller FFT lengths. Thus, the latch-free clock gating technique is applied to power down the unused modules. As a result, the combination of these techniques leads to saving around 30% in power consumption.

REFERENCES

[1] M. Garrido, J. Grajal, M. Sánchez, and O. Gustafsson, "Pipelined radix-2^k feedforward FFT architectures," *IEEE Trans. VLSI Syst.*, vol. 21, no. 1, pp. 23–32, Jan. 2013.

[2] S. He and M. Torkelson, "Design and implementation of a 1024-point pipeline FFT processor," May 1998, pp. 131–134.

[3] Y. O. Park and J.-W. Park, "Design of FFT processor for ieee802.16m MIMO-OFDM systems," in *Int. Conf. Information Comm. Tech. Convergence*, Nov. 2010, pp. 191–194.

[4] M. Garrido, K. K. Parhi, and J. Grajal, "A pipelined FFT architecture for real-valued signals," *IEEE Trans. Circuits Syst. I*, vol. 56, no. 12, pp. 2634–2643, Dec. 2009.

[5] H.-L. Lin, H. Lin, R. Chang, S.-W. Chen, C.-Y. Liao, and C.-H. Wu, "A high-speed highly pipelined 2N-point FFT architecture for a dual OFDM processor," in *Int. Conf. Mixed Design Integrated Circuits Syst.*, Jun. 2006, pp. 627–631.

[6] Y. Chen, Y.-W. Lin, Y.-C. Tsao, and C.-Y. Lee, "A 2.4-Gsample/s DVFS FFT processor for MIMO OFDM communication systems," *IEEE J. Solid-State Circuits*, vol. 43, no. 5, pp. 1260–1273, May 2008.

[7] S.-N. Tang, C.-H. Liao, and T.-Y. Chang, "An area- and energy-efficient multimode FFT processor for WPAN/WLAN/WMAN systems," *IEEE J. Solid-State Circuits*, vol. 47, no. 6, pp. 1419–1435, Jun. 2012.

[8] S. Li, H. Xu, W. Fan, Y. Chen, and X. Zeng, "A 128/256-point pipeline FFT/IFFT processor for MIMO OFDM system IEEE 802.16e," Jun. 2010, pp. 1488–1491.

[9] T. Ahmed, M. Garrido, and O. Gustafsson, "A 512-point 8-parallel pipelined feedforward FFT for WPAN," Nov. 2011, pp. 981–984.

[10] T. Lenart and V. Owall, "Architectures for dynamic data scaling in 2/4/8k pipeline FFT cores," *IEEE Trans. VLSI Syst.*, vol. 14, no. 11, pp. 1286–1290, Nov. 2006.

[11] M. Garrido, J. Grajal, and O. Gustafsson, "'Optimum circuits for bit reversal'," *IEEE Trans. Circuits Syst. II*, vol. 58, no. 10, pp. 657–661, Oct. 2011.

[12] M. Garrido, "Efficient hardware architectures for the computation of the FFT and other related signal processing algorithms in real time," Ph.D. dissertation, Universidad Politécnica de Madrid, 2009.

[13] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*. Prentice-Hall, 1989.

[14] M. Garrido, O. Gustafsson, and J. Grajal, "Accurate rotations based on coefficient scaling," *IEEE Trans. Circuits Syst. II*, vol. 58, no. 10, pp. 662–666, Oct. 2011.

[15] A. Wenzler and E. Luder, "New structures for complex multipliers and their noise analysis," vol. 2, Apr. 1995, pp. 1432–1435.