International Journal of Gender, Science and Technology
ISSN: 2040-0748
UGC Care Group I Journal.
Vol-14 Issue-01 Feb 2025

IMPLEMENTATION OF HIGH SPEED AND POWER EFFICIENT PIPELINED FFT ARCHITECTURE FOR DSP APPLICATIONS

[1]Dr.G CHENCHU KRISHNAIAH, [2]KIRAN KUMAR VAKA, [3]SUNEETHA LALAM,

[1]Professor, Dept. of ECE, Audisankara College of Engineering & Technology (Autonomous), Gudur, Tirupati (Dt),AP, India.

[2]Assistant professor, Dept. of ECE, Audisankara College of Engineering & Technology (Autonomous), Gudur, Tirupati (Dt),AP, India.

[3]PG Scholar, Dept. of ECE, Audisankara College of Engineering & Technology (Autonomous), Gudur, Tirupati (Dt),AP, India.

Abstract

This study introduces a VLSI implementation of a pipelined FFT architecture customized for DSP applications, addressing challenges in real-time processing, high speed, and resource utilization. By exploiting the parallelism of the FFT algorithm and using pipelining techniques, our design achieves high throughput and low latency with minimal area overhead and power consumption. Implemented through VLSI techniques, this architecture can be integrated into dedicated DSP processors or system-on-chip (SoC) designs. It is optimized for area efficiency, making it suitable for resource-constrained applications while maintaining high performance. Extensive simulations and comparisons with existing FFT architectures demonstrate its superior throughput, latency, and lesser delay. Additionally, clock gating strategies are employed to further reduce power consumption. The design, synthesized using Xilinx tools and implemented on an Artix7 FPGA board, was verified for theoretical accuracy using Verilog HDL.

Keywords: Fast Fourier Transform, Radix-2 Butterfly, Pipelining Technique, Clock Gating.

1. Introduction

Specialized processors known as DSPs are meticulously designed microcontrollers engineered to execute the complex mathematical computations essential for real-time signal processing tasks. They are essential in various applications, including telecommunications; audio and video analysis, radar technology, and medical signal processing. DSPs excel in performing operations such as filtering, Fast Fourier Transforms (FFTs), convolution, and signal compression with exceptional efficiency and speed. These processors are characterized by their unique architectures, which support fast arithmetic operations, parallel processing, and low-latency data handling, making them adept at managing the continuous stream of data typical in real-time signal processing scenarios. DSPs incorporate features such as hardware multipliers, specialized instruction sets, and dedicated memory architectures to optimize performance for signal processing tasks. They are capable of executing multiple instructions per cycle and often include SIMD (Single Instruction, Multiple Data) and VLIW (Very Long Instruction Word) capabilities, which enhance their ability to handle large-scale computations and data throughput efficiently. Additionally, DSPs are engineered with power-efficient modes and low-energy consumption features, making them vital for battery-operated and portable devices. Their programmability allows for flexibility in algorithm implementation, making them suitable for a variety of applications and adaptable to evolving technological needs. Furthermore, DSPs frequently support real-time operating systems and offer extensive peripheral interfaces for connecting with other hardware components, enabling seamless integration into larger systems. As a result, DSPs are fundamental to modern digital systems, providing the computational power and efficiency necessary to process and manipulate signals in a real-time environment.

Here is a new pipelined parallel FFT architecture, where initial three stages follow pipelined structure and from fourth stage onwards it follows the parallel pipelined structure. A novel structure of pipelined Fast Fourier Transform is implemented for two radix structures those are radix-2 and radix-2^2.

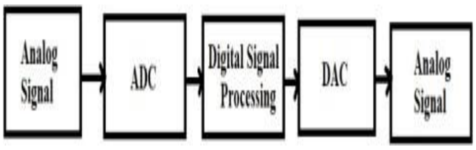


Figure 1: Block diagram of DSP system

The Quick Fourier Transform (FFT) is a vital algorithm in signal analysis that efficiently computes the Discrete Fourier Transform (DFT) and its inverse, facilitating the transformation of signals between the time and frequency domains for easier analysis and manipulation. Implementing FFT using Very-Large-Scale Integration (VLSI) technology is essential for achieving the high-speed, low-power, and area-efficient processing required in modern electronic devices. By utilizing advanced VLSI techniques, Fast Fourier Transform architectures can be optimized to meet the demands of various applications, from communications to medical imaging, providing robust and efficient solutions for complex signal processing tasks.

$$X(k) = \sum_{m=0}^{M-1} x(m).\ W_M^{km}$$

$$W_n = e^{-i2\pi/M}$$
$$0 <= k <= M-1 \wedge 0 <= m <= M-1$$

The radix-2 method is a basic technique utilized for executing the Quick Fourier Transform (FFT) calculations. It efficiently breaks down a discrete Fourier transform (DFT) into smaller DFTs. This method repeatedly divides the input sequence into two halves, processing them recursively to achieve rapid and efficient transformation. Due to its straightforwardness and efficiency, the radix-2 FFT is widely employed in signal processing applications, enabling quick and efficient frequency domain analysis of signals.

2. LITERATURE REVIEW

In [1] this method, present radix-2k feed-forward (MDC) FFT architectures that support any power-of-two parallel sample count and accommodate both Decimation in Time (DIT) and Frequency Decimation (DIF) decompositions. These designs achieve extremely high throughput with fewer hardware resources compared to parallel feedback architectures (MDF), making them perfect for challenging applications and opening new research avenues in feedforward structures. The FFT, crucial for efficiently computing the DFT in signal processing, is implemented in hardware using pipelined architectures for high-performance, real-time processing with low area. Pipelined architectures, including feedback and feed-forward types, are essential for high-throughput applications like OFDM and UWB, addressing challenges such as FFT computation for multiple data sequences and parallel sample processing.

This [2] work describes the implementation of FFT processors using two fused floating-point operations: a two-term dot product and an add-subtract unit. These fused operations optimize the "butterfly" computations in FFT, which involve complex multiplications, additions, and subtractions. Both radix-2 and radix-4 butterfly operations benefit from these fused operations, resulting in FFT processors that are approximately 15% faster and 30% smaller than traditional implementations using high-performance standard cell technology. Moreover, combining these methods provides slightly more precise numerical outcomes by minimizing rounding operations.

In [3] this method Discrete Fourier Transform (DFT) has been replaced by the Fast Fourier Transform (FFT) algorithm, which employs a divide-and-conquer principle to enhance computation speed by leveraging the symmetry of calculations for indices with even and odd values. a significant drawback remains in both FFT and DFT computations rely on complex multipliers that must handle changes in angle at every step due to twiddle factor multiplications, leading to increased hardware complexity, power consumption, and delay.

In [4] this paper, we introduce a novel and efficient FFT algorithm designed for OFDlwDMT applications, along with its pipeline implementation outcomes. The proposed algorithm leverages the radix-4 butterfly unit, enabling it to process data at twice the speed of conventional algorithms. Moreover, the implementation is more area-efficient than traditional radix-4 algorithms because it reduces the number of nontrivial multipliers, similar to the radix-23 algorithm.

The [5] advantages of employing digital convolution for implementing a specific pulse compression radar filter. Given the filter's bandwidth, a straightforward calculation of the necessary computation rate reveals that significant parallel processing would be required with existing integrated circuits. Various DFT computation methods are discussed, with the FFT algorithm being selected due to its regular structure and in-place computation, which facilitate parallel processing. The paper describes the parallelism in the radix-2 pipeline FFT, demonstrating that additional parallel processing is needed to achieve the desired processing rate. By computing n butterflies simultaneously at each stage of the FFT, a family of parallel pipeline FFT processors is developed.

3. EXISTING SYSTEM

The Radix-2^2 algorithm was developed to inherit the simple control structure of Radix-2 but adopt operation and hardware saving technique from Radix-4. This makes Radix-2^2 well suited for VLSI implementation of low power FFT. In [9], by considering the first two decompositions of Radix-2 together.

$$X[k_1 + 2k_2 + 4k_3] = \sum_{n_3=0}^{\frac{N}{4}-1} [H(k_1, k_2, k_3)] . W_{\frac{N}{4}}^{n_3 k_3}$$

Where H (k1, k2, k3) = [A + (− j) (k1+2k2) B].Wn3(k1+2k2)

$$A = x\,[n_3] + (-1)^{k_1}\, x\left[n_3 + \frac{N}{2}\right]$$

$$B = x\left[n_3 + \frac{N}{4}\right] + (-1)^{k_1}\, x\left[n_3 + \frac{3N}{4}\right]$$

Here, A and B are two conventional Radix-2 FFT and H(k1, k2, k2) is also a Radix-2 FFT of A and B with an additional complex multiplier for the twiddle factor Wn3(k1+2k2) N . Using this decomposition, the first two stages of Radix-2 can be transferred into one stage Radix-2^2 consisting of two cascaded Radix-2 FFT. The flow graph of this algorithm is depicted in Fig. 2.2

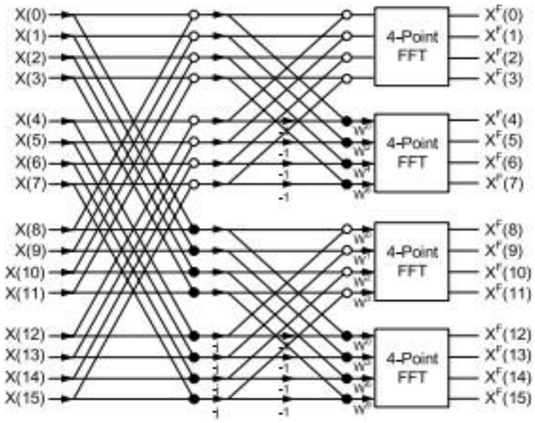


Figure 2: Flow graph of 16-point Radix-4 FFT algorithm.

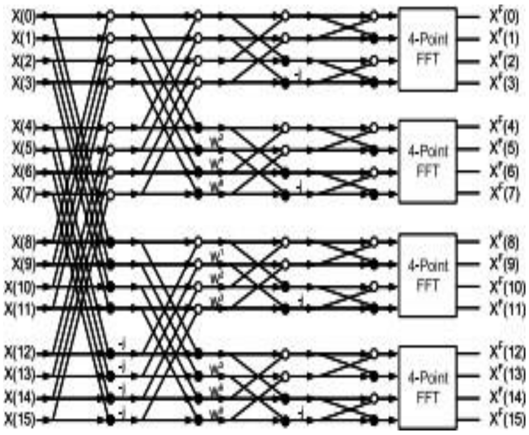


Figure 3: Flow graph of 16-point radix-2^2 FFT algorithm.

Radix-2^2 has a simple architecture of Radix-2 and has the same number of iterated stages (log4N-1) as in Radix-4. In terms of calculating operations and hardware implementation, Radix-2^2 has less complex architecture with a smaller number of multipliers. Therefore, it is very popular for pipeline FFT implementation to reduce the power consumption.

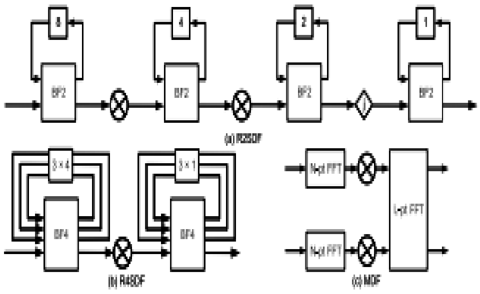


Figure 4: Delay feedback architecture

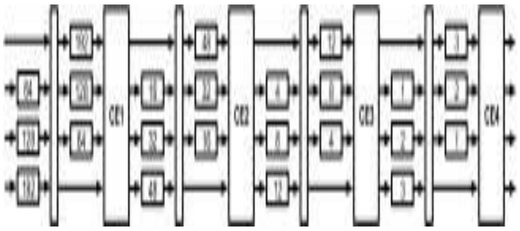


Figure 5: Radix-4 multiple delay commutator

4. PROPOSED SYSTEM

In this study, a thorough design space is introduced exploration of various approximate multipliers, such as the Truncated Multiplier, to assess the potential energy savings and error metrics associated with approximate FFT hardware. By incorporating approximations in the multiplication and addition operations within the butterfly structure, Notable hardware simplifications result in decreases in circuit size, critical path length, and energy usage. Most multipliers and adders implement approximations by focusing on the least significant bits (LSB), allowing for controlled error incorporation in the decimal representation. These approximations are made by varying the operational precision based regarding the number of LSBs used, which typically results in minimal error magnitudes and facilitates higher maximum operating frequencies. Furthermore, to minimize power usage of the circuit, we have implemented clock control in our proposed design. This method can significantly decrease parameters such as power consumption and delay. Addressing power dissipation is a critical design concern in VLSI circuits. Our proposed work primarily focuses on dynamic power dissipation, which is minimized by reducing signal activity within the design. Because the clock network contributes significantly to power dissipation, substantial power savings can be achieved by gating the clock when it is unnecessary.

In many applications, latch based designs are moved to flip flop based designs. By splitting flip flop, we can see two latches from the master slave theorem. In this technique, we can see D flip flop with AND gate. Gated clock goes to high when flip-flop output and clock are in high state otherwise gated clock goes to zero state. That means when clock in sleep mode then gated clock also in zero state.

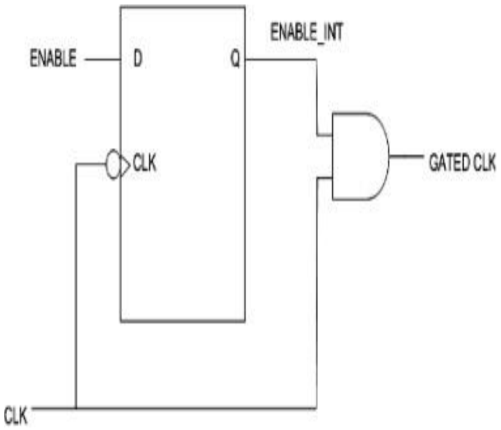


Figure 6: Clock gating using flip-flops

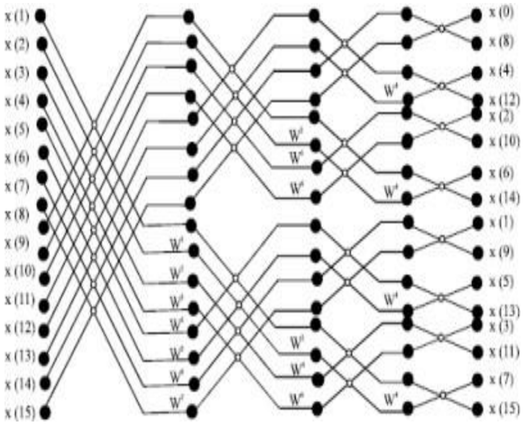


Figure 7: 16 – point DIF FFT structure

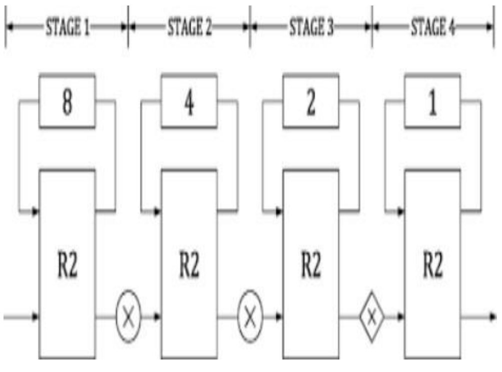
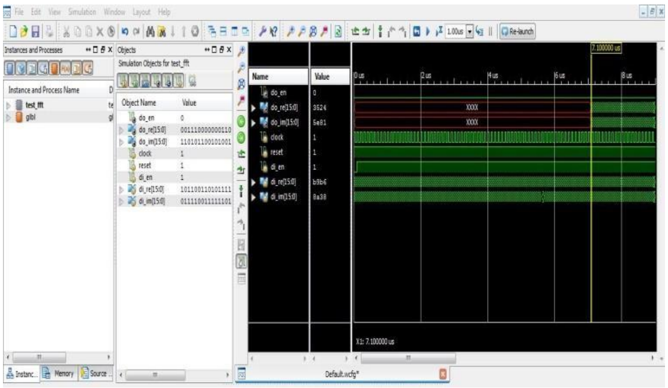


Figure 8: FFT calculation by using radix-2 algorithm

5. RESULTS AND DISCUSSIONS

The experimental results for a 16-point FFT using the radix-2 algorithm were validated for theoretical accuracy with Verilog HDL. The design was implemented on an Artix-7 FPGA board using Xilinx tools.

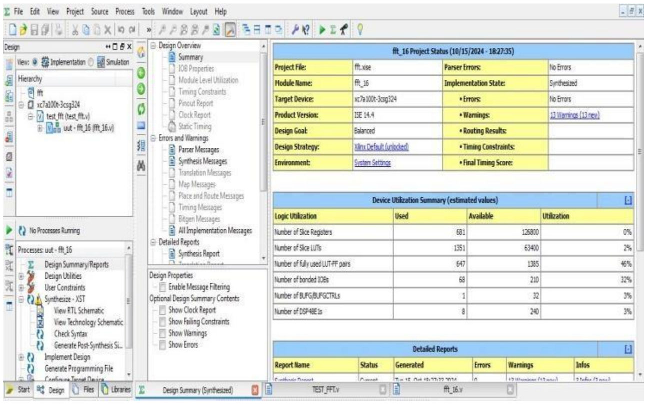


Simulation Objects for test_fft

Object Name	Value
do_en	0
do_re[15:0]	001110000000110
do_im[15:0]	110101100101001
clock	1
reset	1
di_en	1
di_re[15:0]	101100110101111
di_im[15:0]	011110011111101

Name	Value
do_en	0
do_re[15:0]	3624
do_im[15:0]	5e81
clock	1
reset	1
di_en	1
di_re[15:0]	b9b6
di_im[15:0]	8a38

Figure 9: Simulation result of the existing FFT architecture



fft_16 Project Status (10/15/2024 - 18:27:35)			
Project File:	fft.xise	Parser Errors:	No Errors
Module Name:	fft_16	Implementation State:	Synthesized
Target Device:	xc7a100t-3csg324	• Errors:	No Errors
Product Version:	ISE 14.4	• Warnings:	13 Warnings (13 new)
Design Goal:	Balanced	• Routing Results:	
Design Strategy:	Xilinx Default (unlocked)	• Timing Constraints:	
Environment:	System Settings	• Final Timing Score:	

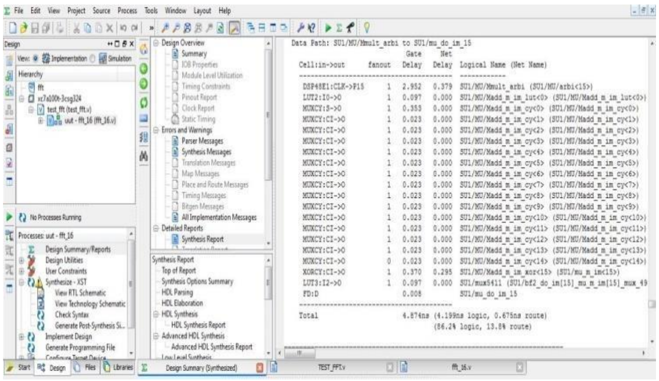
Device Utilization Summary (estimated values)

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	681	126800	0%
Number of Slice LUTs	1351	63400	2%
Number of fully used LUT-FF pairs	647	1385	46%
Number of bonded IOBs	68	210	32%
Number of BUFG/BUFGCTRLs	1	32	3%
Number of DSP48E1s	8	240	3%

Detailed Reports

Report Name	Status	Generated	Errors	Warnings	Infos

Figure 10: Summary report of the existing FFT architecture



```
Data Path: SU1/MU/Mmult_arbi to SU1/mu_do_im_15
                                 Gate     Net
  Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
  ----------------------------------------  ------------
   DSP48E1:CLK->P15      1   2.952   0.379  SU1/MU/Mmult_arbi (SU1/MU/arbi<15>)
   LUT2:I0->O            1   0.097   0.000  SU1/MU/Madd_m_im_lut<0> (SU1/MU/Madd_m_im_lut<0>)
   MUXCY:S->O            1   0.353   0.000  SU1/MU/Madd_m_im_cy<0> (SU1/MU/Madd_m_im_cy<0>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<1> (SU1/MU/Madd_m_im_cy<1>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<2> (SU1/MU/Madd_m_im_cy<2>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<3> (SU1/MU/Madd_m_im_cy<3>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<4> (SU1/MU/Madd_m_im_cy<4>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<5> (SU1/MU/Madd_m_im_cy<5>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<6> (SU1/MU/Madd_m_im_cy<6>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<7> (SU1/MU/Madd_m_im_cy<7>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<8> (SU1/MU/Madd_m_im_cy<8>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<9> (SU1/MU/Madd_m_im_cy<9>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<10> (SU1/MU/Madd_m_im_cy<10>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<11> (SU1/MU/Madd_m_im_cy<11>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<12> (SU1/MU/Madd_m_im_cy<12>)
   MUXCY:CI->O           1   0.023   0.000  SU1/MU/Madd_m_im_cy<13> (SU1/MU/Madd_m_im_cy<13>)
   MUXCY:CI->O           0   0.023   0.000  SU1/MU/Madd_m_im_cy<14> (SU1/MU/Madd_m_im_cy<14>)
   XORCY:CI->O           1   0.370   0.295  SU1/MU/Madd_m_im_xor<15> (SU1/mu_m_im<15>)
   LUT3:I2->O            1   0.097   0.000  SU1/mux5411 (SU1/bf2_do_im[15]_mu_m_im[15]_mux_49
   FD:D                      0.008          SU1/mu_do_im_15
  ----------------------------------------
  Total                      4.874ns (4.199ns logic, 0.675ns route)
                                     (86.2% logic, 13.8% route)
```

Figure 11: Delay report of the existing FFT architecture

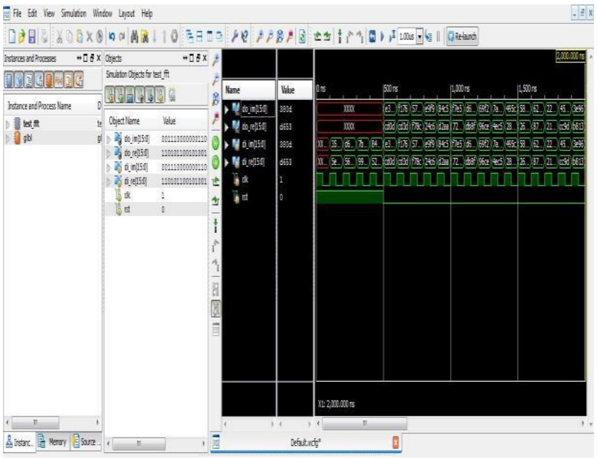
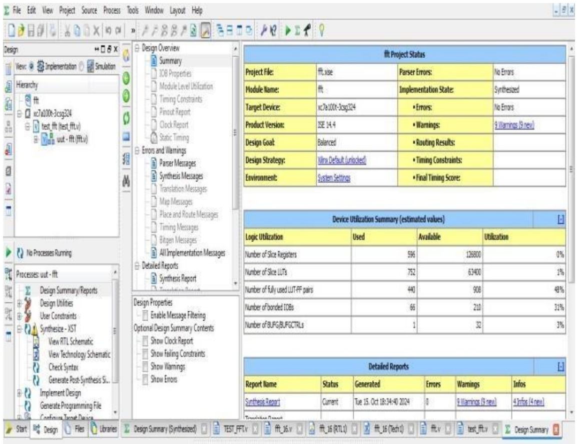


Figure 12: Simulation result of the proposed FFT architecture

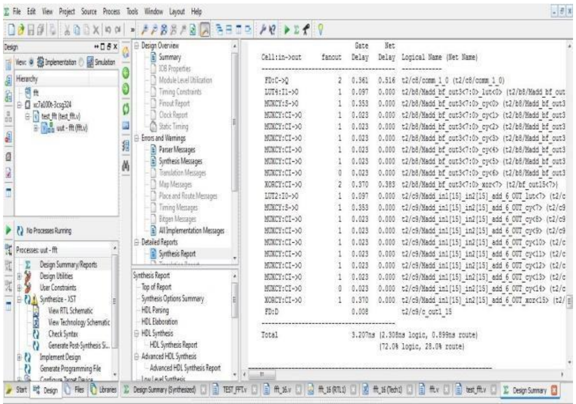


fft Project Status			
Project File:	fft.xise	**Parser Errors:**	No Errors
Module Name:	fft	**Implementation State:**	Synthesized
Target Device:	xc7a100t-3csg324	**• Errors:**	No Errors
Product Version:	ISE 14.4	**• Warnings:**	9 Warnings (9 new)
Design Goal:	Balanced	**• Routing Results:**	
Design Strategy:	Xilinx Default (unlocked)	**• Timing Constraints:**	
Environment:	System Settings	**• Final Timing Score:**	

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	596	126800	0%
Number of Slice LUTs	752	63400	1%
Number of fully used LUT-FF pairs	440	908	48%
Number of bonded IOBs	66	210	31%
Number of BUFG/BUFGCTRLs	1	32	3%

Detailed Reports					
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	Tue 15. Oct 18:34:40 2024	0	9 Warnings (9 new)	4 Infos (4 new)

Figure 13: Summary report of the proposed FFT architecture



```
                          Gate     Net
Cell:in->out      fanout  Delay    Delay  Logical Name (Net Name)
----------------------------------------  ------------
 FD:C->Q               2  0.361    0.516  t2/c8/comm_1_0 (t2/c8/comm_1_0)
 LUT4:I1->O            1  0.097    0.000  t2/b8/Madd_bf_out3<7:0>_lut<0> (t2/b8/Madd_bf_out
 MUXCY:S->O            1  0.353    0.000  t2/b8/Madd_bf_out3<7:0>_cy<0> (t2/b8/Madd_bf_out3
 MUXCY:CI->O           1  0.023    0.000  t2/b8/Madd_bf_out3<7:0>_cy<1> (t2/b8/Madd_bf_out3
 MUXCY:CI->O           1  0.023    0.000  t2/b8/Madd_bf_out3<7:0>_cy<2> (t2/b8/Madd_bf_out3
 MUXCY:CI->O           1  0.023    0.000  t2/b8/Madd_bf_out3<7:0>_cy<3> (t2/b8/Madd_bf_out3
 MUXCY:CI->O           1  0.023    0.000  t2/b8/Madd_bf_out3<7:0>_cy<4> (t2/b8/Madd_bf_out3
 MUXCY:CI->O           1  0.023    0.000  t2/b8/Madd_bf_out3<7:0>_cy<5> (t2/b8/Madd_bf_out3
 MUXCY:CI->O           0  0.023    0.000  t2/b8/Madd_bf_out3<7:0>_cy<6> (t2/b8/Madd_bf_out3
 XORCY:CI->O           2  0.370    0.383  t2/b8/Madd_bf_out3<7:0>_xor<7> (t2/bf_out15<7>)
 LUT2:I0->O            1  0.097    0.000  t2/c9/Madd_in1[15]_in2[15]_add_6_OUT_lut<7> (t2/c
 MUXCY:S->O            1  0.353    0.000  t2/c9/Madd_in1[15]_in2[15]_add_6_OUT_cy<7> (t2/c9
 MUXCY:CI->O           1  0.023    0.000  t2/c9/Madd_in1[15]_in2[15]_add_6_OUT_cy<8> (t2/c9
 MUXCY:CI->O           1  0.023    0.000  t2/c9/Madd_in1[15]_in2[15]_add_6_OUT_cy<9> (t2/c9
 MUXCY:CI->O           1  0.023    0.000  t2/c9/Madd_in1[15]_in2[15]_add_6_OUT_cy<10> (t2/c
 MUXCY:CI->O           1  0.023    0.000  t2/c9/Madd_in1[15]_in2[15]_add_6_OUT_cy<11> (t2/c
 MUXCY:CI->O           1  0.023    0.000  t2/c9/Madd_in1[15]_in2[15]_add_6_OUT_cy<12> (t2/c
 MUXCY:CI->O           1  0.023    0.000  t2/c9/Madd_in1[15]_in2[15]_add_6_OUT_cy<13> (t2/c
 MUXCY:CI->O           0  0.023    0.000  t2/c9/Madd_in1[15]_in2[15]_add_6_OUT_cy<14> (t2/c
 XORCY:CI->O           1  0.370    0.000  t2/c9/Madd_in1[15]_in2[15]_add_6_OUT_xor<15> (t2/
 FD:D                     0.008           t2/c9/c_out1_15
----------------------------------------
Total                     3.207ns (2.308ns logic, 0.899ns route)
                                  (72.0% logic, 28.0% route)
```

Figure 14: Delay report of the proposed FFT architecture

CONCLUSION

This research suggests an innovative design for VLSI architectures of the Quick Fourier Transform (FFT) by leveraging arithmetic approximations. This investigation comprehensively explores the synergetic integration of approximate multiplier architectures within the decimation-in-frequency (DIF) FFT butterfly, examining various levels of approximation in detail. Through extensive design space exploration, a group of approximate arithmetic configurations is elucidated, exhibiting a balanced trade-off between design fidelity and high speed. Finally, the results imply that prolonged multiplier, incorporating distinct approximation techniques, achieves substantial energy savings. The simulation outcomes, along with synthesis reports, have been meticulously analysed, encompassing area and delay metrics obtained via Xilinx tools. Additionally, clock gating methods have been used to further reduce power consumption.

REFERENCES

[1] M. Garrido, J. Grajal, M. A. Sanchez, and O. Gustafsson, “Pipelined radix-2k Feed forward FFT Architectures,” IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, vol. 21, no. 1, pp. 23–32, 2013.

[2] H. M. Saleh and E. Swartzlander, “FFT Implementation with Fused Floating-Point Operations”, IEEE Transactions on Computers, No. 2, Vol. 61, February 2012, pp. 284 – 288.

[3] M Jaber and D. Massicotte “Radix-$2\alpha/4\beta$Building Blocks for Efficient

VLSI's Higher Radices Butterflies Implementation," VLSI Design, vol. 2014, Article ID 690594, 13 pages, 2014.

[4] Jung. Y, Yoon. H, Kim. J, «New Efficient FFT Algorithm Pipeline Implementation Results OFDM/DMT Applications», IEEE Transaction on Consumer Electronics, Vol. 49, No. 1, pp.14- 20, February 2003.

[5] J. A. Johnston, "Parallel pipeline fast Fourier transformer," IEE Proceedings F: Communications, Radar and Signal Processing, vol. 130, no. 6, pp. 564–572, 1983.

[6] M. Garrido, S.-J. Huang, and S.-G. Chen, "Feedforward FFT hardware architectures based on rotator allocation," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 65, no. 2, pp. 581–592, Feb. 2018.

[7] M. Garrido, J. Grajal, M. A. Sanchez, and O. Gustafsson, "Pipelined radix-2k feedforward FFT architectures," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 21, no. 1, pp. 23–32, Jan. 2013.

[8] N. Li and N. P. van der Meijs, "A radix 22 based parallel pipeline FFT processor for MB-OFDM UWB system," in Proc. IEEE Int. SOC Conf. (SOCC), Sep. 2009, pp. 383–386.

[9] ManoharAyinala, Keshab K. Parhi, "Parallel-Pipelined Radix-22 FFT Architecture for Real Valued Signals", Conference Record of the Forty Fourth Asilomar Conference on Signals, Systems and Computers,7-10 Nov,2010.

[10] G. Paim, H. Amrouch, E. A. C. D. Costa, S. Bampi, and J. Henkel, "Bridging the gap between voltage over-scaling and joint hardware accelerator-algorithm closed-loop," IEEE Trans. Circuits Syst. Video Technol., vol. 32, no. 1, pp. 398–410, Jan. 2022.

[11] B. Moons, D. Bankman, L. Yang, B. Murmann, and M. Verhelst, "BinarEye: An always-on energy-accuracy-scalable binary CNN processor with all memory on chip in 28 nm CMOS," inProc. IEEE Custom Integr. Circuits Conf. (CICC), Apr. 2018, pp. 1–4.

[12] ManoharAyinala, Member, IEEE, and Keshab K. Parhi, Fellow, IEEE, "FFT Architecture for Real-Valued Signals Based on Radix-23 and Radix-24 Algorithms", IEEE Transactions on Cir and Sys-I: Regular Papers.

[13] G. Paim et al., "On the resiliency of NCFET circuits against voltage over-scaling," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 68, no. 4, pp. 1481–1492, Apr. 2021.

[14] KodakandlaAbhilash and P.Reena Monica," FFT Architectures for Real Valued Signals based Different Radices Algorithm", Indian Journal of Science and Technology, Vol8(20), DOI:10.17485/ijst/2015/ v8i20/78462, August 2015.

[15] N. Li and N. P. van der Meijs, "A radix 22 based parallel pipeline FFT processor for MB-OFDM UWB system," in Proc. IEEE Int. SOC Conf. (SOCC), Sep. 2009, pp. 383–386.

[16] D. Wold, "Pipeline and parallel-pipeline FFT processors for VLSI implementations," IEEE Trans. Comput., vol. C-33, no. 5, pp. 414–426, May 1984.

[17] S.-I. Cho, K.-M. Kang, and S.-S. Choi, "Implemention of 128-point fast

Fourier transform processor for UWB systems," in Proc. Int. Wireless Commun. Mobile Comput. Conf., Aug. 2008, pp. 210–213.

[18] M. Garrido, J. Grajal, M. A. Sanchez, and O. Gustafsson, "Pipelined radix-2k feedforward FFT architectures," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 21, no. 1, pp. 23–32, Jan. 2013.

[19] M. Garrido, M. Acevedo, A. Ehliar, and O. Gustafsson, "Challenging the limits of FFT performance on FPGAs (Invited paper)," in Proc. Int. Symp. Integr. Circuits (ISIC), Dec. 2014, pp. 172–175.