IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁒ Impact Factor 8.102 ⁒ Peer-reviewed / Refereed journal ⁒ Vol. 12, Issue 4, April 2023

DOI: 10.17148/IJARCCE.2023.124170

FPGA Implementation of Fast Fourier Transform (FFT) Algorithm

Shwetha Kulal[1], Nikitha K[2], Radhika K S[3], Poorvika S[4], Dr. Manjunatha P [5], Mr. Anil Kumar J[6]

Student, Dept. of Electronics and Communication Engineering, JNNCE, Shivamogga, India[1-4]

Professor & Dean Academics, Dept. of Electronics and Communication Engineering, JNNCE, Shivamogga, India[5]

Assistant Professor[6] Dept. of Electronics and Communication Engineering, JNNCE, Shivamogga, India[6]

Abstract: In today's digital signal processing (DSP) world, there is often a need to convert signals between time and frequency domains. FAST Fourier transform (FFT) is a widely used and popular circuit design technique in the communication fields. It is a reduced form of discrete Fourier transform (DFT) in mathematics nature. For this reason, the fast Fourier transform (FFT) has become one of the most important algorithms in the field. The FFT has played a significant role in digital signal processing field, especially in the advanced communication systems, such as orthogonal frequency division multiplexing (OFDM) and asymmetric digital subscriber line. All these systems require that the FFT computation must be high throughput and low latency. Therefore, designing a high-performance FFT circuit is an efficient solution to the abovementioned problems. The proposed architecture is an efficient combined single-path delay commutator-feedback (SDC-SDF) radix-2 pipelined fast Fourier transform architecture, which includes $\log_2 N-1$ SDC stages and 1 SDF stage. The SDC processing engine is proposed to achieve 100% hardware resource utilization by sharing the common arithmetic resource in the time-multiplexed approach, including both adders and multipliers. Thus, the required number of complex multipliers is reduced to $\log_4 N-0.5$, compared with $\log_2 N-1$ for the other radix-2 SDC/SDF architectures. In addition, the proposed architecture requires roughly minimum number of complex adders $\log_2 N+1$ and complex delay memory $2N+1.5\log_2 N-1.5$. Standard FPGA Flow is adapted to implement this project. i.e., right from specification to bit file generation, which is going to be programmed on FPGA. The Work chosen for this project is to Implement Pipelined FFT Architecture using Verilog HD and implemented on the FPGA Spartan6.

Keywords: FFT, SDF, SDC, FPGA.

I. INTRODUCTION

In recent years, as a result of advancing VLSI technology, Orthogonal Frequency Division (OFDM) has received a great deal of attention and been adopted in many new generation wideband data communication systems, spectrum analytics etc. to calculate frequency spectrum and conduct analysis, GPS, LTE and Modern Digital Communication systems. The FFT is one of the most widely used algorithms for calculating the Discrete Fourier Transform (DFT) owing to its efficiency in reducing computation time. Fast Fourier Transform (FFT) is a highly efficient procedure for computing the DFT of a finite series. Some of the applications of Fast Fourier transform include Sound filtering, Image filtering, fast large-integer, and polynomial multiplication, solving difference equations, etc. The main advantage of FFT is speed, which it gets by reducing the number of calculations needed to analyse a waveform. The reason for selecting this project is Fast Fourier Transform (FFT) has been used in a wide range of applications, such as wide-band mobile digital communication system based on OFDM principle.

There are three main types of pipelined FFT architectures: Feedback, feedforward, and serial commutator (SC). First, feedback architectures are characterized by their feedback loops, i.e., some outputs of the butterflies are fed back to the memories at the same stage. which is divided into single-path delay feedback (SDF) and multi-path delay feedback (MDF) architectures. Second, feedforward architectures do not have feedback loops and each stage passes the processed data to the next stage. Among feedforward architectures, multi-path delay commutator (MDC) architectures are the most common ones and process several samples in parallel. Finally, SC FFT architectures are characterized using circuits for bit dimension permutation of serial data. They can process either serial data or parallel data in a continuous flow. The complexity of the DFT direct computation can be significantly reduced by using fast algorithms. One such algorithm is the Cooley Turkey radix-r decimation-infrequency (DIF) FFT, which recursively divides the input sequence into N/r sequences of length r and requires $\log_r N$ stages of computation.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁘ Impact Factor 8.102 ⁘ Peer-reviewed / Refereed journal ⁘ Vol. 12, Issue 4, April 2023

DOI: 10.17148/IJARCCE.2023.124170

The FFT is one of the most widely used algorithms for calculating the Discrete Fourier Transform (DFT) owing to its efficiency in reducing computation time. Fast Fourier Transform (FFT) has been used in a wide range of applications, such as wide- band mobile digital communication system based on orthogonal Frequency Division Multiplexing (OFDM) principle, where the system implementation is only feasible when the equipment complexity and power consumption are greatly reduced by utilizing a real- time FFT transformer to replace the bank of (de)modulators for each individual sub-carriers. FFT, as an efficient algorithm to compute the Discrete Fourier Transform (DFT), is one of the most important operations in modern digital signal Processing and communication systems.

The discrete Fourier transform (DFT), of length N, calculates the sampled Fourier transform of a discrete-time sequence at N evenly distributed points $W_k = 2\pi k/N$ on the unit circle.

The N-point DFT is defined by,

$$X[k] = \sum_{n=0}^{N-1} x(n)e^{(-j2\pi nk)/N}$$

Where k = 0, 1, .and *x(n)* is the input data, and N is any integer power of two.

$$W_N^{nk} = e^{(-j2\pi nk)/N}$$

W_n^{kn} is known as twiddle factor.

The following equation shows the length-N forward IDFT of a sequence X(k):

$$x(n) = (1/N)\sum_{k=0}^{N-1} X[k]e^{(j2\pi nk)/N}$$

Where n = 0, 1, ...N – 1.

It is well known that the radix-2 FFT can be deduced from DFT by factorizing the *N*-point DFT recursively into many 2-point DFTs. Assuming that the input data enters the FFT circuit serially in a continuous flow, the radix-2 MDC and SDF architectures can be directly deduced according to the DFG. The radix-2 MDC architecture is the most direct implementation approach of pipelined FFT, but its hardware utilization is only 50%. Compared, the radix-2 SDF design reduces the required memory size. However, the utilizations of adders and multipliers are still 50%. Besides the basic radix-2 architectures, various high-radix pipelined FFT architectures have also been proposed to address the arithmetic resource utilization problem.

II. METHODOLOGY

2.1 Combined SDC-SDF Radix-2 Pipelined FFT

For single-input data stream, we propose an efficient combined SDC-SDF radix-2 pipelined FFT architecture, and the proposed SDC PE structure can reduce 50% complex multipliers. The proposed FFT architecture consists of 1 pre-stage, log2N − 1 SDC stages, 1 post-stage, 1 SDF stage, and 1 bit reverser.

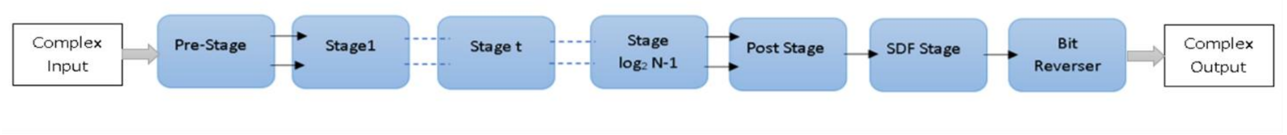


Figure 1: Block diagram of the Combined SDC_SDF FFT Architecture.

A. **Pre-Stage:** The pre-stage shuffles the complex input data to a new sequence that consists of real part followed by the corresponding imaginary part, shown in Fig. 2. The corresponding post-stage shuffles back the new sequence to the complex format.

ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁂ Impact Factor 8.102 ⁂ Peer-reviewed / Refereed journal ⁂ Vol. 12, Issue 4, April 2023

DOI: 10.17148/IJARCCE.2023.124170

B. **SDC:** The SDC stage t (t = 1, 2, …., $\log_2 N - 1$) contains an SDC PE, which can achieve 100% arithmetic resource utilization of both complex adders and complex multipliers. The SDC PE, consists of a data commutator, a real add/sub unit, and an optimum complex multiplier unit.

C. **Post-stage:** The post stage undergoes the date shuffling carried out by the pre-stage.

D. **SDF:** The last stage, SDF stage, is identical to the radix-2 SDF, containing a complex adder and a complex subtracter.

E. **Bit Reverser:** By using the modified addressing method, the data with an even index are written into memory in normal order, and they are then retrieved from memory in bit-reversed order while the ones with an odd index are written in bit-reversed order. Final, the even data are retrieved in normal order. Thus, the bit reverser requires only N/2 data buffer.

Cycle	pre-stage	stage 1	stage 2	stage 3	post-stage
1	0_r, 1_r	-	-	-	-
2	0_i, 1_i	-	-	-	-
3	2_r, 3_r	-	-	-	-
…	…	…	…	…	…
9	8_r, 9_r	-	-	-	-
10	8_i, 9_i	0_r, 8_r	-	-	-
11	10_r, 11_r	0_i, 8_i	-	-	-
12	10_i, 11_i	2_r, 10_r	-	-	-
13	12_r, 13_r	2_i, 10_i	-	-	-
14	12_i, 13_i	4_r, 12_r	-	-	-
15	14_r, 15_r	4_i, 12_i	0_r, 4_r	-	-
16	14_i, 15_i	6_r, 14_r	0_i, 4_i	-	-
17	-	6_i, 14_i	2_r, 6_r	-	-
18	-	1_r, 9_r	2_i, 6_i	0_r, 2_r	-
19	-	1_i, 9_i	8_r, 12_r	0_i, 2_i	0_r,0_i
20	-	3_r, 11_r	8_i, 12_i	4_r, 6_r	2_r,2_i
21	-	3_i, 11_i -	10_r, 14_r	4_i, 6_i	4_r,4_i
…	…	…	…	…	…
32	-	-	-	13_r, 15_r	11_r, 11_i
33	-	-	-	13_i, 15_i	13_r, 13_i
34	-	-	-	-	15_r, 15_i

Figure 2: Data output order of the proposed pipelined architecture for 16-point FFT

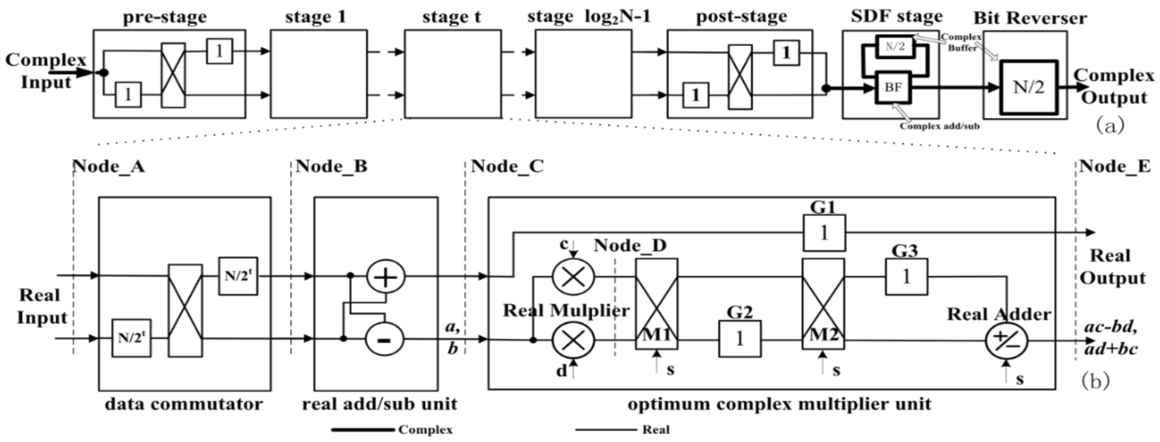


Figure 3: a) Block diagram of the proposed FFT architecture. b) Block diagram of the SDC PE, including a data commutator, a real add/sub unit, and an optimum complex multiplier unit

a (b) means the real (imaginary) part of subtraction result, c (d) means the real (imaginary) part of the twiddle factor. G1, G2, and G3 mean three one-cycle delay elements. The signal s controls the behaviour of two multiplexers (M1 and M2) and the Real Adder. When s is 1 (0), both multiplexers perform "through" ("swap") and the Real Adder performs addition (subtraction) operation.

DOI: 10.17148/IJARCCE.2023.124170

2.2 Single-point DC Processing Engine

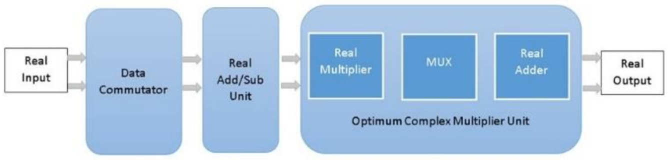


Figure 4: Block diagram of a single SDC processing engine (PE)

The SDC PE, consists of a data commutator, a real add/sub unit, and an optimum complex multiplier unit. In order to minimize the arithmetic resource of the SDC PE, the most significant factor is to maximize the arithmetic resource utilization via reordering the data sequences of the above three units. In the stage t, the data commutator shuffles its input data (Node-A) to generate a new data sequence (Node-B), whose index difference is N/2t , where t is the index of stage. The new data sequence (Node-B) is critical to the real add/sub unit, where one real adder and one real subtracter can both operate on two elements for each input data. The sum and difference results (Node-C) overlap the places of the two input elements. Therefore, it preserves the data sequence, requires only one real adder and one real subtracter. For the optimum complex multiplier unit, its output data sequence (Node-E) should be the same as its input data sequence (Node-C). If so, its output sequence (Node-E), which is also the output sequence of the SDC stage t, can become the direct input data sequence (Node-A) of the SDC stage t+1. Fig 2 illustrates the data sequence of SDC stage 1 of 16-point FFT computation, including the data sequences of the above three units.

2.3 Optimum Complex Multiplier Unit

Cycle	Node_A	Node_B/C	Node_D	G1	G2	G3	s	Real Adder	Node_E	New_Label
1	0_r, 1_r	-	-	-	-	-	-	-	-	-
2	0_i, 1_i	-	-	-	-	-	-	-	-	-
…	…	…	…	…	…	…	…	…	…	…
9	8_r, 9_r	0_r, 8_r	c×8_r ,d×8_r	-	-	-	-	-	-	-
10	8_i, 9_i	0_i, 8_i	c×8_i ,d×8_i	0_r	d×8_r	c×8_r	0	sub	0_r ,c×8_r- d×8_i	0_r* , 8_r*
11	10_r, 11_r	2_r, 10_r	c×10_r ,d×10_r	0_i	c×8_i	d×8_r	1	add	0_i ,c×8_i+d×8_r	0_i* , 8_i*
12	10_i, 11_i	2_i, 10_i	c×10_i ,d×10_i	2_r	d×10_r	c×10_r	0	sub	2_r ,c×10_r- d×10_i	2_r* , 10_r*
13	12_r, 13_r	4_r, 12_r	c×12_r ,d×12_r	2_i	c×10_i	d×10_r	1	add	2_i ,c×10_i+d×10_r	2_i* , 10_i*
14	12_i, 13_i	4_i, 12_i	c×12_i ,d×12_i	4_r	d×12_r	c×12_r	0	sub	4_r ,c×12_r- d×12_i	4_r* , 12_r*
15	14_r, 15_r	6_r, 14_r	c×14_r ,d×14_r	4_i	c×12_i	d×12_r	1	add	4_i ,c×12_i+d×12_r	4_i* , 12_i*
16	14_i, 15_i	6_i, 14_i	c×14_i ,d×14_i	6_r	d×14_r	c×14_r	0	sub	6_r ,c×14_r- d×14_i	6_r* , 14_r*
…	…	…	…	…	…	…	…	…	…	…
24	-	7_i, 15_i	c×15_i ,d×15_i	7_r	d×15_r	c×15_r	0	sub	7_r ,c×15_r- d×15_i	7_r* , 15_r*
25	-	-	-	7_i	c×15_i	d×15_r	1	add	7_i ,c×15_i+d×15_r	7_i* , 15_i*

Figure 3.4: Data Sequence Of the Proposed Stage 1 Of 16-Point FFT

As shown in Fig 3b, it contains 2 multiplexers (M1 and M2), 1.5-word memory (G1, G2, and G3), 2 Real Multipliers and 1 Real Adder. The signal ’s’ controls the behaviour of two multiplexers (M1 and M2): ’through’ or ’swap’. The signal ’s’ also controls the behaviour of the Real Adder, which supports both addition and subtraction operations. For the input couple (0r, 8r) and (0i, 8i) at the Node-C in Fig. 3.4, the sum part data 0r and 0i will directly pass to the delay memory G1, to generate 0r* and 0i* with one cycle delay in consecutive two cycles, while the difference part 8r and 8i will directly enter the Real Multipliers (Node-D) to generate (c × 8r, d × 8r) and (c × 8i, d × 8i) before reordering.

The reordering process is performed as follows.

1. In the first cycle, when 8r comes, the signal ’s’ (s = 1) selects “through”; that is, the up (down) input of the multiplexer (M1 or M2) connects to the up (down) output. Then, the G2 (or G3) would be d × 8r (or c × 8r) in the second cycle.
2. In the second cycle, when 8i comes, the signal s (s = 0) selects “swap”; that is, the up (down) input of the multiplexer (M1 or M2) connects to the down (up) output. Then, the G2 (or G3) would be c × 8i (or d × 8r) in the third cycle. The s will make the Real Adder perform subtraction operation and then c × 8r -d × 8i (8r*) would appear at the Node-E.
3. In the third cycle, the signal s (s = 1) selects “through” for M1 and M2 and chooses addition operation for Real Adder. Then, d × 8r+c × 8i (8i*) would appear at the Node-E. Consequently, the complex result data couple (0r*, 8r*)

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 12, Issue 4, April 2023

DOI: 10.17148/IJARCCE.2023.124170

and (0i*, 8i*) would come out at New-Label (Node-E) with one clock delay in consecutive two cycles. The above mechanism can be iterated by applying to the other couples in the stage 1, e.g., (2r, 10r) and (2i, 10i), and so on. If we carry the above process toward the $\log_2 N$ - 1 stages to completion, we can complete the majority part of the radix-2 FFT computation.

2.4 Hardware Used:

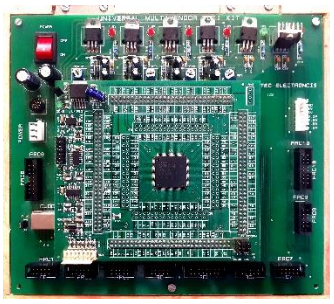


Figure 3.4: Spartan6 FPGA

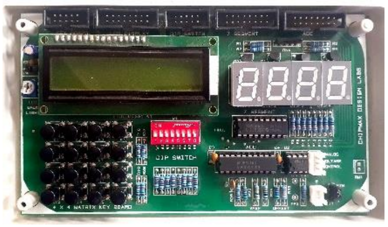


Figure 3.4: External Interface Board

Figure 3.4: USB Programmer for FPGA

FPGAs were invented by Xilinx in 1984. They are structured like the gate arrays form of Application Specific Integrated Circuits (ASIC). However, the architecture of the FPGAs architecture consists of a gate-array-like architecture, with configurable logic blocks, configurable I/o blocks, and programmable interconnect. Additional logic resources such as ALUs, memory, and decoders may also be available. The three basic types of programmable elements for an FPGA are static RAM, anti-fuses, and flash EPROM. Segments of metal interconnect can be linked in an arbitrary manner by programmable switches to form the desired signal nets between the cells. FPGAs can be used in virtually any digital logic system and provide the benefits of high integration levels without the risks of expenses of semicustom and custom IC development. FPGAs give us the advantage of custom functionality like the Application Specific Integrated Circuits while avoiding the high development costs and the inability to make design modifications after production. The FPGAs also add design flexibility and adaptability with optimal device utilization while conserving both board space and system power. The gate arrays offer greater device speed and greater device density. Taking these into consideration, FPGAs are especially suited for rapid prototyping of circuits, and for production purposes where the total volume is low.

Spartan™ 6 devices offer industry-leading connectivity features such as high logic-to-pin ratios, small form-factor packaging, MicroBlaze™ soft processor, and a diverse number of supported I/O protocols. Ideally suited for a range of advanced bridging applications found in consumer, automotive infotainment, and industrial automation.

III. RESULTS AND DISCUSSIONS

Initially, a 16-point FFT was implemented using the normal Butterfly structure method and the results were verified using Xilinx ISE Design Suite, then 16-point Combined SDC-SDF FFT Algorithm is built and verified for its functionality. The RTL synthesis, Device utilization and Timing report were obtained for the 16-point FFT with the proposed architectures using Xilinx tool and the parameters were analysed. The total gate delay and offset was found to be 11.881ns & (3.634ns - 12.056ns) respectively and the Butterfly Structure Algorithm uses many Blocks as the N-point FFT increases. Keeping this into the consideration, we have developed the Verilog code for FFT Computation using Combined Single path Delay Feedback-Commutator (SDF-SDC) Algorithms.

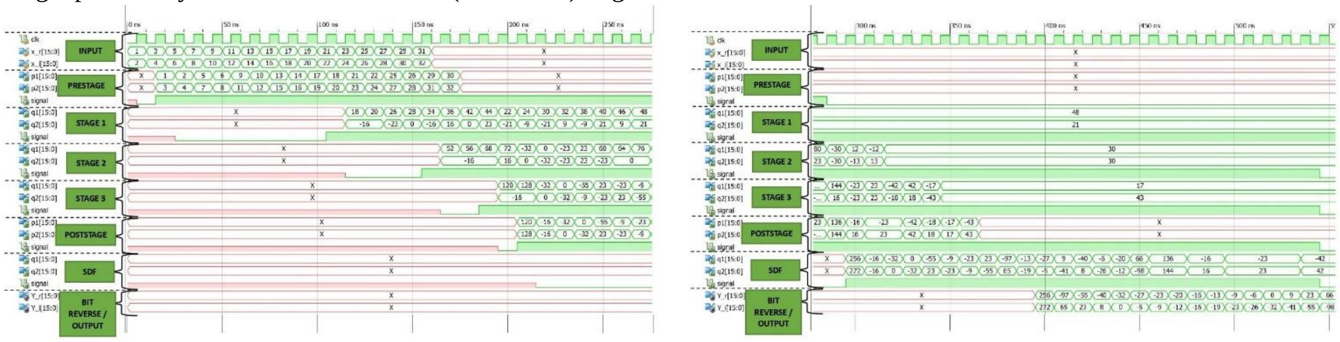
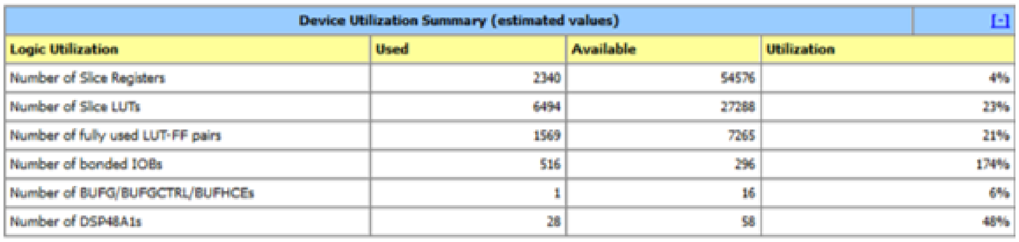


Figure 3.4: Output Waveform16-Point FFT



Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	2340	54576	4%
Number of Slice LUTs	6494	27288	23%
Number of fully used LUT-FF pairs	1569	7265	21%
Number of bonded IOBs	516	296	174%
Number of BUFG/BUFGCTRL/BUFHCEs	1	16	6%
Number of DSP48A1s	28	58	48%

Figure 3.4: Device Utilization

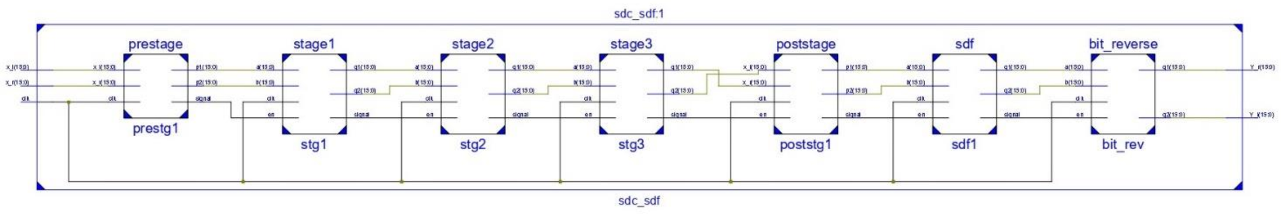


Figure 3.4: Generated RTL Schematic

From the device utilization reports, we can see that the normal butterfly structure algorithm uses more resources whereas the proposed algorithm has shown efficient performance. Hence, we implemented the better algorithm which is efficient in factors like area, timing, and power constraints, on the FPGA kit.

IV. CONCLUSION

The high growth of the semiconductor industry over the past two decades has put Very Large-Scale Integration in demand all over the world. Digital Signal Processing has played a great role in expanding VLSI device area. The FFT is one of the most widely used algorithms for calculating the Discrete Fourier Transform (DFT) owing to its efficiency in reducing computation time. The work chosen for this project is to Implement FFT in FPGA using Verilog HDL. The reason for selecting this project is Fast Fourier Transform (FFT) has been used in a wide range of applications, such as wide- band mobile digital communication system based on orthogonal Frequency Division Multiplexing (OFDM) principle. The objective of this project is to implement a FFT Architecture using Verilog HDL. The Architecture that is going to be implemented is Combined SDC-SDF. Standard FPGA Flow is adapted to implement this project. i.e. right from specification to bit file generation, which is going to be programmed on Spartan6 FPGA. Simulations and Synthesis will be done using Xilinx ISE.

We propose a combined SDC-SDF pipelined FFT architecture which produces the output data in the normal order. The proposed SDC PE mainly reduces 50% complex multipliers, compared with the other radix-2 FFT designs. Therefore, the proposed FFT architecture is very attractive for the single-path pipelined radix-2 FFT processors with the input and output sequences in normal order.

REFERENCES

[1]. Z. Wang, X. Liu, B. He, and F. Yu, “A combined sdc-sdf architecture for normal i/o pipelined radix-2 fft,” IEEE Transactions on very large scale integration (VLSI) systems, vol. 23, no. 5, pp. 973–977, 2014

[2]. M. Garrido, R. Andersson, F. Qureshi, and O. Gustafsson, “Multiplierless unity-gain sdf ffts,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 24, no. 9, pp. 3003–3007, 2016

[3]. X.-Y. Shih, Y.-Q. Liu, and H.-R. Chou, “48-mode reconfigurable design of sdf fft hardware architecture using radix-3 2 and radix-2 3 design approaches,” IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 64, no. 6, pp. 1456–1467, 2017

[4]. C. Ingemarsson, P. Källström, F. Qureshi, and O. Gustafsson, “Efficient fpga mapping of pipeline sdf fft cores,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 25, no. 9, pp. 2486–2497, 2017.

[5]. Y. Chandu, M. Maradi, A. Manjunath, and P. Agarwal, “Optimized high speed radix-8 fft algorithm implementation on fpga,” in 2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI). IEEE, 2018, pp. 430–435