ISSN: 2278-4632 Vol-10 Issue-5 No. 5 May 2020

Design and Implementation of a low latency Radix-8 FFT using FPGA Architecture

¹V.Lavanya, ²B.Kiranmai ¹Department of ECE, M.V.G.R College of Engineering, Vizianagaram ²Department of ECE, Lendi Institute of Engineering and Technology, Vizianagaram ¹lavanyavadda@gmail.com, ²kbabburu@gmail.com

Abstract:

In view of emerging networks low power consumption, low latency and accuracy are the most important issues. To address these issues, optimization is the important aspect that can solve these problems. In the communication channel the low latency is due to the conversion from time domain to the frequency domain at the receiver. The conversion of the time domain to frequency domain can be obtained by using FFT/IFFT which involves butterfly radix methodology. In this paper, a new algorithm is developed to reduce the computation operation in Radix-8 FFT implementation and which intern reduces the area, power consumption and latency. This optimized Radix-8 provides better performance compared with the previous Radix architectures.

Keywords: Radix, FFT, Low latency, Twiddle factor, Complex Multiplication

I. INTRODUCTION

The efficient computations and analysis of the spectrum are obtained in the fields of communications through conversion from time domain to frequency domain and which can be achieved by DFT/FFT [1]. Mainly FFT has been used in OFDM systems, Wireless communications, GPS [2]. In FFT algorithm, the computation time and complexity can be reduced by applying logarithmically [3]. In the literature there are number of algorithms were proposed to reduce the complexity and time taken to reduce the computation time [7].

The proposed algorithms for the Digital signal processing are performed excellent on MATLAB toolboxes [4]. In this context, a digital signal processor which converts time domain signals to frequency domain are the crucial and needs new approaches and applications for reducing the latency. So optimized and low computation FFT implementation is very essential for reducing the latency in DSP processors [5]. The VLSI technology has given enormous changes for implementing the DSP hardware design with low power, area and reduced latency [6]. In the present day scenario, the required and the desirable throughput are to be achieved with reduced time and power [8].

ISSN: 2278-4632 Vol-10 Issue-5 No. 5 May 2020

In this context, to reduce the design complexity and time delay in higher radix FFT implementation because the higher radix requires the higher hardware design and complexity depends on the length of FFT implementation. A new algorithm has been proposed in this paper to achieve less complexity in design and reduced latency for conversion in 8 point FFT implementation.

II. Algorithm for proposed architecture

The radix format is the most efficient design for computing the DFT by using FFT. In this section the existed systems for Radix-2, Radix4, Radix-8 are discussed and also the proposed optimized Radix-8 architecture algorithm is presented here.

II.I RADIX-2 ALGORITHM

The basic fundamental method which was proposed by the Cooley-Tukey for the computation of the DFT is butterfly method. The radix-2 is that which requires inputs 2 and outputs2 at the end [9]. The equations 1 and 2 explain the computation of DFT using butterfly method.

The above equations 1 and 2 are implemented with adder, subtractor and twiddle factor. The butterfly diagram for 8 point FFT in DIF (decimation in frequency) is shown in figure1



Figure 1: Radix-2 8-point butterfly flow diagram

www.junikhyat.com

ISSN: 2278-4632 Vol-10 Issue-5 No. 5 May 2020

The figure 1 shows the butterfly diagram of the 8point FFT and the mid stage represents the twiddle factor, computed for converting the values.

II.II RADIX-4 ALGORITHM

The Radix-4 algorithm has computations for 4 inputs and will have 4 outputs at the end. The Radix -4 computations are represented by these following equations.

$$\begin{aligned} x(4k) &= \sum_{n=0}^{\left(\frac{N}{4}\right)-1} \left(\left[x(n) + x(n + \left(\frac{N}{4}\right)) + x\left(n + \left(\frac{N}{2}\right)\right) + x\left(n + \left(\frac{3N}{4}\right)\right) \right] w_N^0 \right) \right] w_{\frac{N}{4}}^{kn} \right) &--3 \\ x(4k+1) &= \sum_{n=0}^{\left(\frac{N}{4}\right)-1} \left(\left[x(n) - j * x(n + \left(\frac{N}{4}\right)) - x\left(n + \left(\frac{N}{2}\right)\right) + j * x\left(n + \left(\frac{3N}{4}\right)\right) \right] w_N^0 \right) \right] w_{\frac{N}{4}}^{kn} \right) &--4 \\ x(4k+2) &= \sum_{n=0}^{\left(\frac{N}{4}\right)-1} \left(\left[x(n) - x(n + \left(\frac{N}{4}\right)) + x\left(n + \left(\frac{N}{2}\right)\right) - x\left(n + \left(\frac{3N}{4}\right)\right) \right] w_N^0 \right) \right] w_{\frac{N}{4}}^{kn} \right) &--5 \\ x(4k+3) &= \sum_{n=0}^{\left(\frac{N}{4}\right)-1} \left(\left[x(n) + j * x(n + \left(\frac{N}{4}\right)) - x\left(n + \left(\frac{N}{2}\right)\right) - j * x\left(n + \left(\frac{3N}{4}\right)\right) \right] w_N^0 \right) \right] w_{\frac{N}{4}}^{kn} \right) &--6 \end{aligned}$$

The equations from 3 to 6 are representing the computational complexity for calculating the twiddle factor for Radix4 and the butterfly diagram is shown in the figure with DIF.



Figure 2: radix 4 4-point FFT butterfly diagram

To obtain the 8 point FFT for Radix 4 the further stages are coupled with Radix-2.

II. III RADIX-8 ALGORITHM

The Radix-8 algorithm requires the 8 inputs for computation and provides 8 outputs at the end. The computational complexity is very high as the Radix increases and that is shown in these following equations.

$$\begin{split} X(k) &= \sum_{m=0}^{N-1} x(8m) W_N^{k(8m)} + \sum_{m=0}^{N-1} x(8m + 1) W_N^{k(8m+1)} + \sum_{m=0}^{N-1} x(8m + 1) W_N^{k(8m+1)} + \sum_{m=0}^{N-1} x(8m + 1) W_N^{k(8m+1)} + \sum_{m=0}^{N-1} x(8m + 1) W_N^{k(8m+3)} + \sum_{m=0}^{N-1} x(8m + 1) W_N^{k(8m+4)} + \sum_{m=0}^{N-1} x(8m + 1) W_N^{k(8m+5)} + \sum_{m=0}^{N-1} x(8m + 1) W_N^{k(8m+6)} + \sum_{m=0}^{N-1} x(8m + 7) W_N^{k(8m+7)} \\ &= DFT_{N/8}^{[x(8n)]} + W_N^{k} DFT_{N/8}^{[x(8n+1)]} + W_N^{k} DFT_{N/8}^{[x(8n+2)]} + W_N^{k} DFT_{N/8}^{[x(8n+2)]} + W_N^{k} DFT_{N/8}^{[x(8n+4)]} + W_N^{k} DFT_{N/8}^{[x(8n+5)]} + W_N^{k} DFT_{N/8}^{[x(8n+6)]} + W_N^{k} DFT_{N/8}^{[x(8n+6)]} + W_N^{k} DFT_{N/8}^{[x(8n+7)]} + W_N^{k} DFT_{N/8}^{[x(8n+6)]} + W_N^{k} DFT_{N/8}^{[x(8n+7)]} + W_N^{k} DFT_{N/8}^{[x(8n+6)]} + W_N^{k} DFT_{N/8}^{[x(8$$



Fig 3: radix 8 8-point flow diagram

III Proposed Radix-8 algorithm architecture

The computational complexity and delay increases along with the higher Radix FFT implementation but higher throughput will be achieved with higher Radix FFT. In this proposed algorithm, to decrease the computational time and increase the throughput, the standard Radix -8 algorithms is analyzed and this can be achieved by assigning base to 8.

In the proposed algorithm, an N point DFT is separated into groups of iteratively associated Radix 'R' point transformation and the input x(n) be is powers of R. For Radix 8 algorithm 'R' value is 8 and the Decimation in frequency (DIF) Radix-8 FFT can be obtained by repetitively partitions DFT with 'R' point transformation i.e into eight length DFT with individual eighth sample . In this context the distributed outputs of smaller FFTs are can be again utilized for computing the overall outputs because of this the computational complexity reduces and the latency will be reduced effectively.

Hence the complex computations that are carried at the first stage of FFT implementation will be pushed to the final stage and thus reducing the multiplication time required, area, the delay and the power utilization at the same time it improves the maximum throughput.



Figure 4: Proposed optimized Radix 8 FFT flow diagram

The figure 4 shows the proposed Radix 8 FFT diagram, in this algorithm the stage 1 has less computations as it takes the input values and processes to the stage 2. With this less computation the input values are given to the stage 2 so the computational speed increases and the design complexity reduces. Hence reduction in multiplications and complexity, improves the system design, power consumption and latency.

The proposed architecture of Radix-8 algorithm consists of the output as complex and real terms. In this proposed system the given 8 bit inputs are taken as real data. This data is further preceded with using butterfly Radix-2 implementation which reduces the delay path. This Radix 2 module is calculated and every time instances it will be called for small computations. Hence the simplified computations continued up to the final stage calculations which improve faster system calculations and efficient final value in frequency domain.

In this proposed system the twiddle factor have been calculated will be stored in the register this stored value in the register multiplied with the stage 1 calculations. This

ISSN: 2278-4632 Vol-10 Issue-5 No. 5 May 2020

process will be continued for the each until the final stage is obtained. Hence the final stage computations in the 8 point Radix-8 FFT reduced and hence it is a high speed improved technique.

IV. Result





🚴 profft - [D:/naidu/profft/profft.xpr] - Vivado 2014.2														
File Edit Flow Tools Window	Layo	ut	View Run Help								0	ζ≁ Search c	ommands	
	> >	101	🇠 🎉 🔼 🛛	🤪 💾 Default La	ayout	- X	۰ 🔍		(∀) 100	ns 👻 🤘		¢,		Ready
Flow Navigator «	< 8	leha	vioral Simulation	- Functional - sim_	1 - profft_tb									×
Q 🔀 🖨	50		🥺 profft.v 🗙 🛛 🥺	profft_tb.v x	😸 Untitled	2 ×								82×
	bject												1,00	0.000 ns *
Project Planager Project Settings			Name	Value	10 mc		1200 mc		1400 mc		1600 mc		1900 mg	
Add Sources				01100010						(<u>1000</u>		V 1100		V 1000
IR Catalog	8		- III × 1[7:0]	01001100	0000	1000	00000	1010	1100	0010	1101	V 0011	1011	V 0100
- IF Catalog	erti	5	x2[7:0]	10011111	0000	0000	0111	1110	0101	1010	0000	X 1111	1100	0011
▲ IP Integrator	P.C	÷		10001111	0000	0110	0011	0111	1011	1001	1010	1000	1010	0011
ở Create Block Design	<u>@</u>		▲ 🗤 😽 x4[7:0]	11111000	0000	0000	1110	0001	0010	1001	0001	0100	1011	0111
Dpen Block Design			Image: Second state of the second state of	10110111	0000	1000	1000	1000	0110	0001	1100	1101	0010	0001
🍓 Generate Block Design	8		N ⊡	10011111	0000	0110	1111	1111	0110	0000	0010	0111	0000	1111
	l S	1	🖆 🖽 📲 x7[7:0]	01011100	0000	0001	1100	1100	0000	0101	0000	1000	0111	1101
Simulation	-		🖄 🖬 📲 y0[7:0]	00011110	0000	0110	1001	0101	0001	0000	1011	0100	1111	1001
Simulation Settings		8	🚰 🖽 y1[7:0]	11011101	0000	1101	0110	1100	0010	1100	0100	1110	1101	0001
(Run Simulation	1		📃 🖽 📲 y2[7:0]	11101000	0000	0010	0100	1101	1011	0111	0001	1101	1101	1000
4 PTI Applyric			📊 🖬 📲 y3[7:0]	11110101	0000	0111	0011	1110	1001	0010	0001	X 1111	0100	1001
Open Elaborated Design		13		10111110	0000	1000	0101	1110	1010	0110	0110	0000	1111	0101
p open baborated besign			ol ⊡¶{ y5[7:0]	11101011	0000	1100	1001	1100	0000	0001	1101	0110	× 1111	1001
▲ Synthesis		1	1 🖬 📲 y6[7:0]	00101111	0000	1110	1000	0011	0000	0001	0111	X 1001	0010	X 0011
🍪 Synthesis Settings			📰 🖽 y7[7:0]	11111110	0000	0011	<u> </u>	1110	1110	1001	1000	X 1010	<u> </u>	<u> </u>
Run Synthesis														
Open Synthesized Desig														
 Implementation 														
Implementation Settings														
Run Implementation														
Open Implemented Desig			4 1	<	• •									Ť,
Program and Debug	-		Td Console 🔎											
													Sim	n Time: 1 us 📋 🔡
🚱 🦂 🚞 🖸 🌔 🝌 🔉 🏧 🕎 - 🕪 🖬 😼 -														

Fig 6: Simulation result of the proposed system

www.junikhyat.com

ISSN: 2278-4632 Vol-10 Issue-5 No. 5 May 2020

🔀 Eile Edit View Project Source Process I	ools <u>W</u> indow Layout <u>H</u> elp	р								- 8 ×
∽ c X ∩ ∩ X ∽ c	» 🏓 🏓 🕅 🥦 🏓 🛃	🔊 🗟 🗄 🗖 🖻	₽ K? ►	> 🗶 📌 🛛 🖓						
Design ↔ 🗆 🗗 🗙 🚺	27									
Wex: ● 鎌 implementation ○ 璽 Smudaton Hierarchy □ ○ existing □ ○ xr/a100e-3csg324 □ ○ V test, fft (test, fft.x) □ ○ West, out - fft (fft.x) □ ○ □ ○										
No Processes Running		and anticipation		Ť					-	
Processes: uut - fft 🔥 Å		and the second second			and the second second				and the second	
Image: State		<u>P</u> P	1	<u>p</u>					Ì	
View RTL Schematic View Technology Schematic			Ŷ	<u>P</u>	₽ 				1	
Generate Post-Synthesis Si Implement Design Generate Programming File										
Start In: Design Terret Device	Design Summary (Synthesized)		fft.v	× 8	fft (RTL2)		fft (Tech2)	X		
View by Category	in the sin									+ □ # X
Design Ob	jects of Top Level Block					Proper	ties of Instance: ff	t .		
Instances A Pins	▲ Si	gnals		Name			▼ Value			^
B	ŧ	⊢ <mark>å</mark> ft	1	Type SHREG_MIN_SIZE	Nec		fft:1 2			v
📳 Console 🔕 Errors 🔔 Warnings 🙀 Find in	Files Results View by Categ	gory								[1004,6520]

Fig 7: Technology schematic of the existing system



Fig 8: Technology schematic of the proposed system

ISSN: 2278-4632 Vol-10 Issue-5 No. 5 May 2020

Eile Edit View Project Source Process Jools Window	v Layout Help								- 5 ×
🖓 🗐 🕼 🐇 🕼 🗋 🗙 🕪 🕞 🖉	ø ø 🏓 🖻 🗾 🕞 🗉 T	1 🖙 🔑 K? 🕨	· 🛯 🖈 🖉						
Design ++ 🗆 🗗 🗙 📄 Design Or	verview A			fft Project St	atus				^
View:	OB Properties	Project File:	existing.xise	Parser Errors:			No Errors		
Hierarchy	Aodule Level Utilization	Module Name:	fft	Implen	plementation State:		Synthesized		
existing	iming Constraints	Target Device:	xc7a100t-3csg3	124 •	Errors:		No Errors		
B V test_fft (test_fft.v)	Jock Report	Product Version:	ISE 14.4		Warnings:		17 Warnings (17 new)		
🔲 🕀 🕅 uut - fft (fft.v)	tatic Timing	Design Goal:	Balanced	Routing Resu		lts:			
30 P.	'arser Messages	Design Strategy:	Xilinx Default (u	nlocked)	Timing Constraints:				
M	ynthesis Messages	Environment:	System Settings	1 · · · ·	Final Timing S	icore:			
	ranslation Messages Map Messages								
	lace and Route Messages		Device	Utilization Summary (estin	ated values)				Ŀ
Bitgen Messages		Logic Utilization		Used	Available		Utilization		
No Processes Running	II Implementation Messages	Number of Slice Regist	ers	12	12 126800 25 63400				0%
Processes: uut - fft	Reports wathesis Report	Number of Slice LUTs		2!					
👷 🛛 Design Summary/Reports		Number of fully used LUT-FF pairs		12		25		483	
Design Utilities Design Prope Design Prope Design Prope	arties la Marcaga Filtaring	Number of bonded IOE	36	16	16 210				
- Optional Desi	ign Summary Contents	Number of BUFG/BUFG	CTRLS		1	32			3%
View RTL Schematic	/ Clock Report								
Check Syntax	Warnings	Detailed Reports							[-1]
Generate Post-Synthesis Si	/ Errors	Report Name	Status	Generated	Errors	Warnings		Infos	
Generate Programming File		Synthesis Report	Current	Mon 23. Sep 01:06:03 2019	0	17 Warnings (s (17 new) 0		
Configure Target Device		T	() (n = n)			1	-		~
Start 🖤 Design 🛄 Hies 🛄 Libraries 🔬 Design Summa	ary (Synthesized)	fft.v	្យ fft (R1L2) ្រ	I K fft (lech2)		Design Summary	×		
View by Category									⇔⊡₽×
Design Objects of Top Le	zvel Block			Prop	erties of Inst	ance: fft			
Instances ^ Pins	Signals	A	A Name Value						^
			Type SUPEG MINI SIZE		fft:1				
			CUBEC EVERACE NICE		L UTC				~
Console 🥨 Errors 🤽 Warnings 🕅 Find in Files Results	View by Category								





Fig 10: Power report of the existing system

profft - [D:/naidu/profft/profft.x]	or] - Vivado 2014.2	×
File Edit Flow Tools Window	Layout View Help	
🖊 🗁 🔛 🗠 🗤 🖬 🛏 🗙	🥙 🔊 🕨 📩 🚰 🥝 🧐 🐝 🔽 🦃 😬 Default Layout 💿 👻 👘 🌸 🔪 Implementation Com	plete
Flow Navigator <<	Implemented Design * - xc7a100tcsg324-2 (active)	×
🔍 🛣 📾	Power_power_1 _ B	\times
Concernation Direction	🔍 🖾 🚔 🛨 🔤 Summary	
Generate block Design	Settings	
 Simulation 	Summary (49): Power analysis from implemented netist. Activity derived	
Simulation Settings	Dynamic: 49.169 W (99%)	
(Run Simulation	Hierarchica Total On-Chip Power: 49.29 W	
A RTL Analysis	Signals (0.) Junction Temperature: 30.0 C 99% 98% ■ Logic: 0.296 W (1%)	
Open Elaborated Design	□ Logic (0.24 Effective d)A: 4.6 *C/W	
	Device Static: 0.121W (1%)	
- Synthesis	Confidence level:	
Bun Synthesis		
Open Synthesized Desig		
Implementation		
Mig Implementation Settings		
Run Implementation		
Implemented Design Constraints Wiggerd		
Constraints Wizard		
Beport Timing Summ		
Report Clock Networ		
Report Clock Interac		
Report DRC		
Report Noise		
Report Utilization	Dependence of the second secon	
Report Power	📟 Td Console 🖉 Messages 🔄 🔩 Log 🔛 Reports 🕕 Design Runs 🔛 Power	
	Tuesday, February	25, 2020
		PM
		2020

Fig 11: Power report of the proposed system

ISSN: 2278-4632 Vol-10 Issue-5 No. 5 May 2020

	5 10											-
D 🖉 🗟 🕼 🐇 📓 🖓 🖸 🖸 🗙 🖸	1	» / / / / / / / 🔁 🗾 🕞 🗖 🛛	1 -	۶ 🖈 🗈 📌 🖇	2							
esign ↔⊡ & ×	03	Design Overview				fft Project	Status (02/18)	2020 -	17:52:14)			
👔 View: 🖲 🎆 Implementation 🔿 🎆 Simulation	-	IOB Properties	Project	File:	fft, xise		Par	Parser Errors:			No Errors	
Hierarchy	0	Module Level Utilization	Module	Name:	fft		Im	lemen	tation State		Synthesized	
a - 🖸 fft	۲	- Timing Constraints	Treast	Davidant	vc7a100t-2cma224			- Foregoin			No Francisco	
🚔 🖻 🛄 xc7a100t-3csg324	0	Pinout Report	Target	Device:	xc7a100t-3csg	324		• ст	ors:		NO Errors	
in Man fft (fft.v)		Clock Report	Product	Version:	ISE 14.7			•Wa	rnings:		No Warnings	
	_	Frrors and Warnings	Design	Goal:	Balanced			• Ro	uting Results:			
	30	- 🖻 Parser Messages	Design	Strategy:	Klinx Default (L	inlocked)		• Tin	ning Constraints:			
2	(4)	- 🔄 Synthesis Messages	Environ	ment:	System Setting	ŝ		• Fin	al Timing Score:			
2	-	Translation Messages										
		Place and Route Messages			-				-			
		- Timing Messages			Device	Julization 5	ummary (estim	ated v	alues)	1		1
		- D Bitgen Messages	Logic U	tilization		Used		Avai	able	Utilizati	on	
No Processes Running		All Implementation Messages	Number	of Slice LUTs			139		6340	0		0
C Processes: fft		Synthesis Report	Number	of fully used LUT-FF pairs			C		13	9		0
🖞 🚽 🚬 Design Summary/Reports		×	Number	of bonded IOBs			128	1	21	0		60
📄 🗄 🎽 Design Utilities		Design Properties	-					÷				
User Constraints		Optional Design Summary Contents	-				1 10 1					
Be C Implement Design		- Show Clock Report			1	Deta	alled Reports	ports				L
Generate Programming File		Show Failing Constraints	Report	Name	Status	Genera	ted		Errors War	nings	Infos	
B Configure Target Device		Show Warnings	Synthesi	s Report	Current	Tue 18, F	eb 17:52:12 2020		0 0		0	
Analyze Design Using Chipscope			Translatio	on Report								
			Map Rep	ort								
			01	Jn. 1. n	-	-					1	
Implementation		Module Level Utilization Module Level Utilization Pinout Report Clock Report Clock Report Static Timing Parer Messages Synthesis Messages		Delay: Source: Destination: Data Path: x2 Cell:in->ou	4.: x2: y0: <1> to yi t fi	205ns (I <1> (PAI <7> (PAI 0<7>	Gate Delay De	Net	= 12)	(Net Na	ame)	
No Processes Running Processes fit Design Summary/Reports Design Utilities		Map Messages Place and Route Messages Place and Route Messages Bitgen Messages Detailed Reports Synthesis Report Synthesis Report	s 🗸	IBUF:I->0 LUT2:I0->0 MUXCY:S->0 XORCY:CI-> LUT3:I2->0 LUT4:I3->0 MUXCY:S->0 XORCY:CI-> LUT2:I0->0	0	2 1 4 1 1 1 1 1	0.001 0. 0.097 0. 0.353 0. 0.370 0. 0.097 0. 0.097 0. 0.353 0. 0.353 0. 0.370 0. 0.370 0.	384 000 309 295 000 379 000	<pre>x2_1_IBUF (x2 Madd_y1_lut<? Madd_y1_cy<12 Madd_y1_cy<23 Madd_s203 (Ma Madd_s203 (Ma Madd_s20_lut</pre> Madd_s20_cy<0 Madd_s20_xov	1_IBU > (Madd (Madd > (¥1_ :0>5 (Ma :0>5 (Ma :0>5 (Ma :0>5 (S :0>5 (Ma :0>_5 (S) :0>_5 (Madd)	F) d_y1_lut<1>) 2_OBUF) 2_OBUF) 3) add_s20_lut< add_s20_cy<0 s20<6>) d_y0_lut<6>)	(0>5))>5)
No Processes Running Processes: fft Design Utilities User Constraints Oronstraints		Map Messages M	s > <	IBUF: I->0 LUT2: I0->0 MUXCY: S->0 XORCY: CI-> LUT3: I2->0 MUXCY: S->0 XORCY: CI-> OBUF: I->0 Total	0 0 0	2 1 1 1 1 1 1 1 0 1	0.001 0. 0.9353 0. 0.353 0. 0.370 0. 0.097 0. 0.353 0. 0.370 0. 0.000 0. 0.0000 0. 0.000 0. 0.0000 0. 0.0000 0. 0.0000 0. 0.0000 0. 0.0000 0. 0.00000 0. 0.0000 0. 0.0000000000	384 000 309 295 000 379 000 279 .558 0.8%	x2_1_IBUF (x: Madd_y1_lut <br Madd_y1_lut <br Madd_s20_s(x) Madd_s20_s(x) Madd_s20_lut- Madd_s20_lut- Madd_s20_lut- Madd_y0_lutMadd_y0_lutMadd_y0_oxMadd_y0_oxMadd_y0_oxMadd_y0_oxMadd_y0_s0r(y0_7_0BUF (y0ns logic, 1.64	1 IBU (Madd) (Madd (Madd) (Madd (Madd) (M) (M) (M) (M) (M) (M) (M) (M	<pre>F)</pre>	:0>5))>5)
No Processes Running Processes Iff Design Summary/Reports Design Unitides Synthesize - XST Configue Target Dexice Configue Target Dexice Configue Target Dexice Start Dif Configue Target Dexice Start Dif Configue Target Dexice Start Dif Configue Target Dexice Dif Con	e ies [Map Messages M	s v (IBUF:I-30 IIBUF:I-30-90 IUIX:II-30-90 NUXCY:S-30-90 IUI3:I2-30 NUXCY:S-30	O O 	2 1 4 1 1 1 1 1 0 1	0.001 0.353 0.370 0.097 0.097 0.353 0.370 0.353 0.353 0.370 0.353 0.370 0.353 0.370 0.353 0.370 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.357 0.355 0.355 0.357 0.355	384 000 309 295 000 000 279 000 279 .558 0.8%	<pre>x2_1IBUF(w: Madd y1_utc) Madd y1_evc1 Madd =203 HW Madd =203 HW Madd =20 Lut- Madd =20 Lut- Madd y0_evc3 Wadd y0_evc3 y0_7_0BUF(y1 ns logic, 1.64 logic, 39.2%</pre>	1 IBU > (Madd > (y1 - ;)(Madd > (y1 - ;)(Madd > (y1 - ;)(Madd > (y1 - ;)(Madd - (Madd - (Madd - (Madd - (Madd - (Y0 - ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7>) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)(<7) ;)()()()()()()()()()()()()()()()()()()(<pre>F) 4 y1_lut<l>) y1_cy<l> 2 05UF) 3) add s20_cy add s20_cy<0 z20<6>) 4 y0_lut<6>) y0_cy<6>) _05UF)</br></l></l></pre>	(0>5)



	LUT	Delay	Memory usage
Existing	138	4.955 ns	385292 kb
(8- point radix 2)			
Proposed	139	4.205 ns	386348 kb
(Optimized Radix 8)			

Table 1: comparison of Radix-8 FFT architectures

V. Conclusion and Future work

The proposed Radix 8 optimized system technique improves the high throughput and area required. Power consumption and delay can be reduced. The FFT implementation with reduced latency implementation has enormous applications in the wireless communications where the OFDMA implementation is purely depends on FFT /IFFT modules. The latency in the wireless communications can be reduced with this proposed algorithm.

ISSN: 2278-4632 Vol-10 Issue-5 No. 5 May 2020

Further this work can be enhanced with an Application Specific Integrated Chip (ASIC) for reducing the latency and improving the throughput in wireless communication systems

References

[1] T. Widhe, "Design of Efficient Radix-8 Butterfly PEs for VLSI," IEEE Trans. Audio and Electroacoustics, vol. AU-17, pp. 109 - 119, 2013.

[2] M. Rajasekhar, K. Manjula, "Design and Simulation of 512 Point FFT using RADIX-8 Algorithm," International Journal of Applied Sciences, Engineering and Management ,ISSN 2320 – 3439, Vol. 05, No. 04, pp. 30 – 33, 2016.

[3] LihongJia, YonghongGao and HannuTenhunen, "Efficient VLSI Implementation of Radix-8 FFT Algorithm," IEEE 7 803 -5 582-2/99, 2009.

[4] Mingxi Sun, LiyuTian and Dongmin Dai, "Radix-8 FFT Processor Design Based on FPGA," 5th International Congress on Image and Signal Processing, 2012.

[5] Shousheng He and Mats Torkelson," Designing Pipeline FFT Processor for OFDM (de) Modulation" URSI International Symposium on Signals, Systems, and Electronics. Conference Proceedings, PISA, ITALY, 2008.

[6] Saad Bouguezel, M. Omair Ahmad," A New Radix-2=8 FFT Algorithm for Length-q 2m DFTs" IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, VOL. 51, NO. 9, 2004.

[7] Kala S, Nalesh S, ArkaMaity, S K Nandy," High Throughput, Low Latency, Memory Optimized 64K Point FFT Architecture using Novel Radix-4 Butterfly Unit", IEEE International Symposium on Circuits and Systems, 2013.

[8] MengjunLi,Yongxin Zhu, Xu Wang, Tian Huang, Weida Chen, Bin Liu, Yishu Mao," Evaluation of Variable Precision Computing with Variable Precision FFT Implementation on FPGA" International Conference on Field-Programmable Technology, 2016.

[9] MounirArioua, Said Belkouch, Mohamed Agdad," VHDL implementation of an optimized 8-point FFT/IFFT processor in pipeline architecture for OFDM systems," International Conference on Multimedia Computing and Systems, 2011.

[10] Johan Lofgren and Peter Nilsson," On Hardware Implementation of Radix 3 and Radix 5 FFT Kernels for LTE systems" International Conference on Multimedia Computing and Systems, 2011.

[11] Vinay Kumar M, David Selvakumar A, Sobha P M," Area and Frequency optimized 1024 point Radix-2 FFT Processor on FPGA", International Conference on VLSI Systems, Architecture, Technology and applications, 2016.