IMPLEMENTATION OF PIPELINED RADIX-2 FFT USING SDC AND SDF ARCHITECTURE

Deepika Hiremath
4th Semester, M.Tech Digital Electronics & Communication Systems
Department of Electronics & Communication Engineering
PES University, Bengaluru-560085
dpkhrmth@gmail.com

Mrs. Rajeshwari. B
Associate Professor
Department of Electronics & Communication Engineering
PESIT , Bengaluru-560085
rajeshwari@pes.edu

***Abstract*- Implementation of the Pipelined Radix_2 FFT using Single path Delay Commutator and Single path Delay Feedback (SDC & SDF) architecture is presented. This architecture includes 3 stages of the SDC and 1 stage of SDF. This approach makes use of less hardware resources and also shares the same resources as it is pipelined architecture. Thus reducing the number of complex multipliers and the adders used in the architecture as compared with other Radix_2 SDC and SDF architecture when used alone.**

***Keywords*- Radix_2 FFT, Pipelined Architecture, Single_path Delay Commutator (SDC), Single_path Delay Feedback (SDF).**

I. INTRODUCTION

Technologies in the VLSI have become more advanced and have also contributed a great platform for the programmable Digital Signal Processing (DSP) devices, so that they can be used in wider range and are also available at the affordable prices. Digital signal processors can be used either as application-specific or general-purpose. Application-specific chips are designed mainly to perform one specific function, which has more advantages in terms of accuracy, speedy operations. The examples of the application purpose chips are Digital filters, Fast Fourier Transform chips. Algorithms which are used for fast computation of DFT and IDFT are known as FFT algorithm. These algorithms use power of 2 points and exploit the periodic nature of complex exponent $e^{\frac{-j2\pi kn}{N}}$ occurring in the DFT and IDFT equations.

Techniques used in DSP architecture to increase their speed of operation are Parallelism & Pipelining. There are 2 types of Pipelined architecture basically: Delay Commutator (DC) and Delay Feedback (DF). Depending on the number of input these 2 are further divided into SDC (Single-path Delay Commutator) MDC (Multiple-path Delay Commutator) SDF (Single-path Delay Feedback) MDF (Multiple-path Delay Feedback).

In [9], Author has chosen MDC architecture, power consumption and the hardware resources used is more, even though the speed is achieved at the desired rate. In [8], author has proposed SDF architecture to reduce complex adder by 50% and also produce the output in normal order. Multiplier utilization remains 50%. In [10], the Author has proposed SDC architecture using matrix factorization, then converting them into expressions using functional operators and implementing on the hardware. In [1], authors have proposed the architecture combining both SDC & SDF architectures in order to reduce the hardware resources, but it has some delay associated with it, so this architecture has been modified to reduce 50% less multiplexer than the existing and also creating less nodes so as to reduce the delay and hardware resources.

In brief this paper aims to design a high performance Radix-2(R2) FFT circuit which has high throughput (in terms of latency), small chip area and reduces the hardware resources more than 50% as compared to the SDC or SDF architecture when used alone. The architecture also gives normal output sequences. To implement pipelined radix-2 FFT using SDC-SDF architecture, it requires 3 SDC stages, 1 SDF stage, and 1-BR (Bit Reverser). SDC stages helps in achieving 100% hardware utilization of adders and multipliers used in this architecture. SDF stage is required to re-order the data and also reduces the memory and adders. This architecture produces normal output order with the help of bit reverser.

II. PIPELINED FFT ARCHITECTURE

A. FFT

Fast Fourier Transform is the algorithm which is used to perform the fast calculations of discrete fourier transforms. FFT works in the fashion by decomposing the bigger block into smaller once. This project deals with the 16 point FFT. The equation of "N-point DFT" is given by:

$$X(K) = \sum_{n=0}^{N-1} \left(x(n) \times w^{\frac{nk}{N}} \right)$$

where k =1,2,3,4,……..,N-1,X(k) is complex o/p, x(n) is complex i/p data,.N = 16 (16 point DFT), $w^{\frac{nk}{N}}$ is the twiddle factor (co-efficient).

The "Data Flow Graph" (DFG) of 16-pointt R_2 FFT is shown below. The DFG shown below is decimation in frequency (DIF). DIF has the advantages like it reduces the multipliers used in the architecture to half as compared to decimation in time (DIT) and also the adders used will be less. This method is easy to implement on hardware.

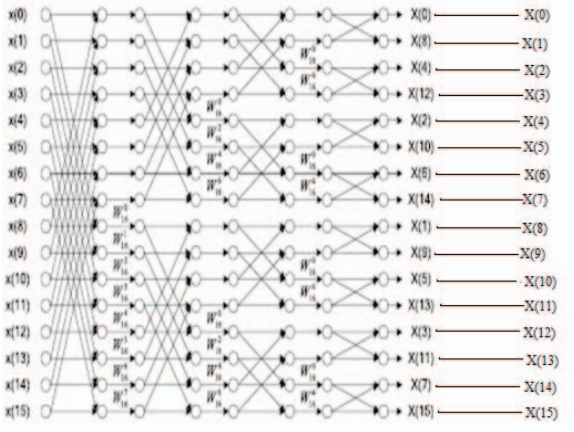
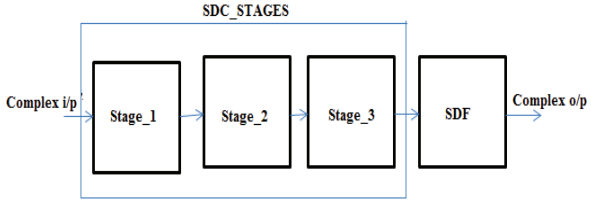


Fig 1. 16-point FFT

The above architecture is arrived by solving the DFT equation. So the proposed architecture will arrive at the same sequence as shown in the above Fig 1.

B. PROPOSED PIPILINED ARCHITECTURE

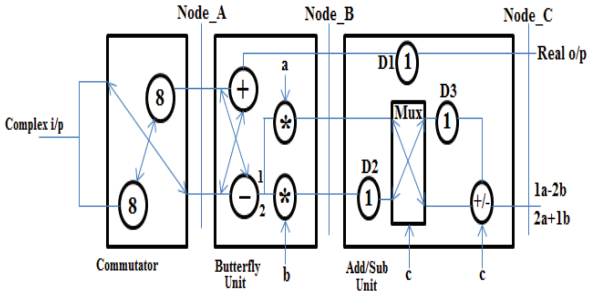
The combined SDC-SDF architecture is shown in Fig 2, (a) is the basic architecture and (b) SDC stages (c)SDF stage



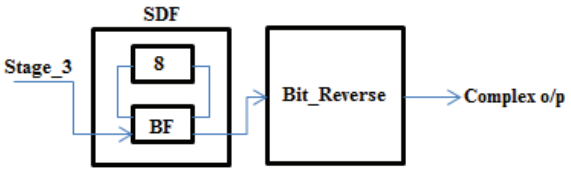
(a)

Basic Pipelined architecture shown in Fig 2(a) consists of three stages of "SDC" architecture, one "SDF stage" and one "Bit Reverse stage". The commutator shown in Fig 2(b) of the "SDC" stage_1 divides the complex input into 2 parts i.e. real input followed by the imaginary input, as it processes single input. So in the first cycle only real output is arrived and in the next cycle imaginary output is obtained. Stages 1-3 are the SDC stages helps in achieving 100% hardware utilization of arithmetic resources (adders and multipliers).

The Node_C combines the obtained real and imaginary output into complex output, in order to obtain the normal output order. In fig 2(c) SDF stage and Bit Reversal stage is shown. SDF stage consists of both adder and subtraction in butterfly unit and 1 memory to store the even output sequence. This is then subjected to Bit Reversal stage which maps to the respective orderly output sequences. The output sequence of this architecture is shown in the Table I.



(b)



(c)

Fig 2. Basic Pipelined architecture (a) Architecture of both SDC and SDF stages, (b) SDC architecture of Stage_1 (c) SDF stage.

The data sequence of the pipelined 16-point FFT computation is shown in Table I. The complex input data at cycle m are (number−rl, number−ig), where number−rl and number−ig (number=0, 1, 2.3,4,5,6,7,8,9,10,11,12,13,14,15) represents the real and the imaginary terms, respectively.

SDC stage consists of the butterfly unit which is adding and subtracts multiplier part which consists of the real twiddle factor multiplier and the imaginary twiddle factor multiplier. Then this output is subjected to the mux and the real add/sub part which takes care of combining the output into complex which was fed individually, 3 delays are used so that the outputs should not be overlapped.

In the stage 1, the commutator divides its comp i/p data into a new data (Node−A), whose difference is equal to 8i, where i is the index of stage and N is equal to 16, i.e. commutator divides the input by 2 (N/2), so for Stage 2:N=8 & for Stage 3: N=4. This data is fed to the add/sub unit, where either 1 adder or 1 subtraction can be performed for each input, the difference result is then sent to twiddle

part multipliers (Node_B).Node_B is then subjected to multiplexer part which is again sent to add/sub part (Node_C).

The SDC Stage contains one multiplexers (Mux), three delay memories (D1, D2, and D3), two (Real) Rl Multipliers and one (Real) Rl Adder as shown in the Fig 2(b). The signal 'c' controls the operation of the multiplexer (Mux): through or swap. The term a and b represents the real and odd part of the twiddle factor.

. The results of the Node_A are subjected to the add and sub part, wherein the numbers undergoing addition are directly subjected to output Node_C and the number undergoing subtraction will be first multiplied with twiddle factor and then subjected to Mux part. The calculation is performed in following steps.

1. The real input enters, depending on the signal "c" i.e for first cycle c=0, mux will just pass the inputs to outputs in same sequence it appears and then performs addition.
2. Imaginary input enters in the 2nd cycle, the signal c (c = 1) selects "swap" i.e. the down input connects the up output or up input of the multiplexer (Mux) connects to the down output. The signal c controls the Real Adder, and subtraction will be carried out for this cycle, thus output appears at the Node−C.
3. In the next cycle, the signal c chooses 0(c=0) selects "through" for mux and addition operation is carried out for the real adder.

The signal 'c' basically controls the operation of the Real Adder and the Mux at a time.

TABLE I

DATA FLOW OF THE STANDARD PIPELINED ARCHITECTURE

Cycle	Complex i/p	Stage 1	Stage 2	Stage 3	Complex o/p
1	0_rl,0_ig	-----	-----	-----	-----
2	1_rl,1_ig	-----	-----	-----	-----
-	----	-----	----	-----	------
8	7_rl,7_ig	------	------	------	------
9	8_rl,8_ig	------	------	------	------
10	9_rl,9_ig	0_rl,8_rl	------	------	-----
11	10_rl,10_ig	0_ig,8_ig	------	------	-----
12	11_rl,11_ig	2_rl,10_rl	------	------	------
13	12_rl,12_ig	2_ig,10_ig	-----	------	-----
14	13_rl,13_ig	4_rl,12_rl	------	-----	-----
15	14_rl,14_ig	4_ig,12_ig	0_rl,4_rl	------	------
16	15_rl,15_ig	6_rl,14_rl	0_ig,4_ig	------	------
17	------	6_ig,14_ig	2_rl,6_rl	------	------
18	------	1_rl,9_rl	2_ig,6_ig	0_rl,2_rl	------
19	------	1_ig,9_ig	8_rl,12_rl	0_ig,2_ig	0_rl,0_ig
20	------	3_rl,11_rl	8_ig,12_ig	4_rl,6_rl	2_rl,2_ig
21	------	3_ig,11_ig	10_rl,14_rl	4_ig,6_ig	4_rl,4_ig
24	------	7_rl,15_rl	1_ig,5_ig	……	…..
---	------	-----	------	…….	…..
28	------	-------	9_ig,13_ig	…….	….
--	------	------	…..	…….	…..
34	-------	------	-----	----	15_rl,15_ig

III. COMPARISION OF VARIOUS ARCHITECTURES

1. MEMORY LATENCY & HARDWARE COMPARISION

Table II gives the brief summary on Memory, Latency & Hardware resources used in different SDC & SDF architectures and its clear from the Table II that the memory and latency used in SDF and SDC architecture alone is more as compared to the proposed.

TABLE II

NO	ARCHITECTURE	MEMORY	LUTs	LATENCY	COMPLEX MULTIPLIER	REAL MULTIPLIER	COMPLEX ADDER	SWITCH	MUX	TOTAL
1	R2SDC	3N-2	32701	256	8	14	31	-	-	53
2	R2 SDF	2N-1	21824	31	4	-	16	-	-	20
3	R2SD2F	(7N-4)/3	640	36	4	-	4	40	-	48
4	R2^3SDF	2N-1	18688	31	2	2	16	-	-	20
5	R2SDC-SDF[1]	2N+1.5(Log2N -1.5)	672	20	2		4	-	2	8
6	**PROPOSED**	**2N(Log2N-1.5)**	**512**	**10**	**2**	**-**	**4**	**-**	**1**	**7**

The hardware resources utilized in the Proposed Architecture is more than 50% less as compared to lone pipelined architecture and also considerably less than the No[5] Architecture shown in the Table II.

2. COMPARING THE HARDWARE RESOURCES ON FPGA

It's clear from the Table III that the Proposed Design has given the best results when implemented on SPARTAN 6 FPGA board and the LATENCY for the proposed architecture is **56nsec** which is considerable less as compared to the existing Pipelined Architectures.

TABLE III

SL. NO	ARCHITECTURE	DEVICE	16-BIT ADDER	16-BIT MULTIPLIERS	TOTAL
1	R2SDC	SPARTAN 6	31	22	53
2	R4SDF	SPARTAN 6	38	12	50
3	R2 SDC_SDF	VIRTEX 5	25	14	39
4	R2SDF	SPARTAN 3	22	12	34
5	**PROPOSED**	**SPARTAN 6**	**14**	**12**	**26**

IV. CONCLUSION

Proposed architecture of Radix-2 FFT reduces more than 50% of the complex multipliers and adders as compared to other lone Pipelined R-2 FFT architecture.
This architecture has also shown significant reduction in the delay, i.e. less latency. The proposed architecture also exhibits normal output order.

V. SCOPE FOR FUTURE WORK

The proposed architecture is designed for 16 bit inputs, it can be further extended for 64, 256, 1024, 4096 bit inputs. The intermediate delay while performing the FFT computation is more so the improvisation can be done to eliminate the delay by modifying the architecture. The architecture designed for FFT has shown good results on FPGA board, which can further be implemented and tested for the Digital Filters, complex calculations, successive approximations, etc.

.

.

REFERENCES

[1] Zeke Wang, Xue Liu, Bingsheng He, and Feng 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, May 2015

.[2] C. Cheng and K. K. Parhi, "High throughput VLSI architecture for FFT computation*," IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 54, no. 10, pp. 339–344, Oct. 2007

[3] T. Sansaloni, A. Perez-Pascual, V. Torres, and J. Valls, "Efficient pipeline FFT processors for WLAN MIMO-OFDM systems," *Electron. Lett.,* vol. 41, no. 19, pp. 1043–1044, Sep. 2005.

[4] T. Lenart and V. Owall, "Architectures for dynamic data scaling in 2/4/8K pipeline FFT codes*," IEEE Trans. Very Large Scale Inegr. (VLSI) Syst.*, vol. 14, no. 11, pp. 1286–1290, Nov. 2006.

[5] K. Yang, S. Tsai, and G. Chuang, "MDC FFT/IFFT processor with variable length for MIMO-OFDM systems*," IEEE Trans. Very Large Scale Inegr. (VLSI) Syst*., vol. 21, no. 4, pp. 720–731, Apr. 2013.

[6] M. Ayinala, M. Brown, and K. Parhi, "Pipelined parallel FFT architectures via folding transformation," *IEEE Trans. Very Large Scale Inegr. (VLSI) Syst.*, vol. 20, no. 6, pp. 1068–1081, Jun. 2012

[7] E. H. Wold andDespain, "Pipeline and Parallel-Pipeline FFT processors for VLSI Implementation," *IEEE Trans. Comput.*, vol. C-33, no. 5, pp. 414–426, May 1984

.[8] Y. N. Chang, "An efficient VLSI architecture for Normal I/O order Pipeline FFT Design," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 55, no.12, pp.1234–1238, Dec. 2008

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

[10] A. Cortes, I. Velez, and J. F. Sevillano, "Radix *rk* FFTs: Matricial representation and SDC/SDF pipeline implementation," *IEEE Trans. Signal Process.*, vol. 57, no. 7, pp. 2824–2839, Jul. 2009