Volume 6, Issue 4, April 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Review Paper on Radix-2 Fast Fourier Transform using Real Value Data

[1]Deepenti Dawande, [2]Geetesh Wagadre, [3]Jitendra Ku. Mishra
[1] M. Tech. Scholar, [2] HOD & Associate Professor, [3] Associate Professor
[1, 2, 3] Elect. & Comm. Department, Patel Institute of Engineering and Science, Bhopal, (M.P.) India

Abstract— with the advent of new technology in the fields of VLSI and communication, there is also an ever growing demand for high speed processing and low area design. It is also a well-known fact that the chip area and maximum combinational path delay (MCPD) unit forms an integral part of processor design. Due to this regard, high speed and low area architectures become the need of the day. A fast Fourier transform (FFT) is any fast algorithm for computing the DFT. The development of FFT algorithms had a tremendous impact on computational aspects of signal processing and applied science. The decimation-in-time (DIT) fast Fourier transform (FFT) very often has advantage over the decimation-in-frequency (DIF) FFT for most real-valued applications, like speech/image/video processing, biomedical signal processing, and time-series analysis, etc., since it does not require any output reordering.

Index Terms— FFT, Decimation in Time, Decimation in Frequency, real Value data

I. INTRODUCTION

Digital signal processing (DSP) is the mathematical manipulation of an information signal to modify or improve it in some way. It is characterized by the representation of discrete time, discrete frequency, or other discrete domain signals by a sequence of numbers or symbols and the processing of these signals [1].

The goal of DSP is usually to measure, filter and/or compress continuous real-world analog signals. The first step is usually to convert the signal from an analog to a digital form, by sampling and then digitizing it using an analog-to-digital converter (ADC), which turns the analog signal into a stream of numbers. However, often, the required output signal is another analog output signal, which requires a digital-to-analog converter (DAC). Even if this process is more complex than analog processing and has a discrete value range, the application of computational power to digital signal processing allows for many advantages over analog processing in many applications, such as error detection and correction in transmission as well as data compression. DSP algorithms have long been run on standard computers, as well as on specialized processors called digital signal processor and on purpose-built hardware such as application-specific integrated circuit (ASICs). Today there are additional technologies used for digital signal processing including more powerful general purpose microprocessors, field-programmable gate arrays (FPGAs), digital signal controllers (mostly for industrial apps such as motor control), and stream processors, among others [2-3]. The FFT is one of the most commonly used digital signal processing algorithm. Recently, FFT processor has been widely used in digital signal processing field applied for OFDM, MIMO-OFDM communication systems. FFT/IFFT processors are key components for an orthogonal frequency division multiplexing (OFDM) based wireless IEEE 802.16 broadband communication system; it is one of the most complex and intensive computation module of various wireless standards physical layer (ofdm-802.11a, MIMO-OFDM 802.11, 802.16,802.16e) [4].

Some folded pipeline architectures have been proposed for the computation of RFFT [3], [4], where butterfly operations are multiplexed into a small logic unit. The structures in [3] and [4] could provide adequate throughput for some applications but the storage complexity of those structures continues to be very high. A few in-place architectures have also been proposed for RFFT using specialized packing algorithms [5], [6]. Memory-conflict for read/write operation is found to be the major challenge in the design of algorithms and architectures for in-place computation [7]. Recently, an in-place architecture and conflict-free memory addressing scheme have been proposed for continuous processing of RFFT [8].

The FFT algorithms are classified into two broad categories, namely, the decimation-in-time (DIT) and the decimation-infrequency (DIF) algorithms. The key differences between the two are shown in Fig. 1. In case of DIF algorithm (Fig. 1(a)), the input samples are fed to the computing structure in their natural order, while the output is generated in bit-reversed order. On the other hand, in case of DIT algorithm (Fig. 1(b)), the input samples need bit-reversal reordering before being processed, while the output FFT coefficients are generated in natural order. In different RFFT applications such as image and video processing, biomedical signal processing, and time-series analysis etc., the complete input sequence is generally available together at the same time for the FFT computation. The DIT RFFT has an advantage over the DIF form for these applications, since a DIT RFFT structure need not wait for the arrival of input samples but can produce the outputs as soon as those are computed.

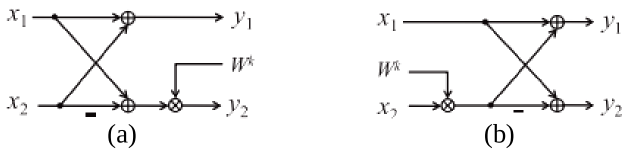

Figure 1: (a) DIF FFT butterfly (b) DIT FFT butterfly

II. LITERATURE REVIEW

Pramod Kumar Meher et al. [7], the decimation-in-time (DIT) fast Fourier transform (FFT) very often has advantage over the decimation-in-frequency (DIF) FFT for most real-valued applications, like speech/image/video processing, biomedical signal processing, and time-series analysis, etc., since it does not require any output reordering. Besides, the DIT FFT butterfly involves less computation time than its DIF counterpart. In this paper, we present an efficient architecture for the radix-2 DIT real-valued FFT (RFFT). We present here the necessary mathematical formulation for removing the redundancies in the radix-2 DIT RFFT, and present a formulation to regularize its flow graph to facilitate folded computation with a simple control unit. We propose here a register-based storage design which involves significantly less area at the cost of a little higher latency compared with the conventional RAM-based storage. The address generation for folded in-place DIT RFFT computation with register-based storage is challenging since both read and write operations are performed in the same clock cycle at different locations. Therefore, we present here a simple formulation of address generation for the proposed radix-2 DIT RFFT structure. The proposed structure involves 61% less area and 40% less power consumption than those of [8], on average, for FFT sizes 16, 32, 64, and 128. It involves 70% less area-delay product and 57% less energy per sample than those of the other, on average, for the same FFT sizes.

Manohar Ayinala et al. [8], this brief presents a novel scalable architecture for in-place fast Fourier transform (IFFT) computation for real valued signals. The proposed computation is based on a modified radix-2 algorithm, which removes the redundant operations from the flow graph. A new processing element (PE) is proposed using two radix-2 butterflies that can process four inputs in parallel. A novel conflict-free memory-addressing scheme is proposed to ensure the continuous operation of the FFT processor. Furthermore, the addressing scheme is extended to support multiple parallel PEs. The proposed real-FFT processor simultaneously requires fewer computation cycles and lower hardware cost compared to prior work. For example, the proposed design with two PEs reduces the computation cycles by a factor of 2 for a 256-point real fast Fourier transform (RFFT) compared to a prior work while maintaining a lower hardware complexity. The number of computation cycles is reduced proportionately with the increase in the number of PEs.

Shashank Mittal et al. [9], fast Fourier Transform (FFT) is one of the most basic and essential operation performed in Software Defined Radio (SDR). Therefore designing a universal, reconfigurable FFT computation block with low area, delay and power requirement is very important. Recently it is shown that Bruun’s FFT is ideally suited for SDR even when operating with higher bit precision to maintain same NSR. In this paper, authors have proposed a new architecture for Bruun’s FFT using a distributed approach for incrementing the number of bits (precision) with successive stages of FFT. It is also shown that proposed architecture further reduces the hardware requirement of Bruun’s FFT with negligible changes in its NSR. The proposed design makes Bruun’s FFT, a better option for most practical cases in SDR. A detailed comparison of Bruun’s traditional and proposed hardware architectures for same NSR is carried out and results of FPGA and ASIC implementations are provided and discussed.

Byung G. Jo et al. [10], the paper proposes a new continuous-flow mixed-radix (CFMR) fast Fourier transform (FFT) processor that uses the MR (radix-4/2) algorithm and a novel in-place strategy. The existing in-place strategy supports only a fixed-radix FFT algorithm. In contrast, the proposed in-place strategy can support the MR algorithm, which allows CF FFT computations regardless of the length of FFT. The novel in-place strategy is made by interchanging storage locations of butterfly outputs. The CFMR FFT processor provides the MR algorithm, the in-place strategy, and the CF FFT computations at the same time. The CFMR FFT processor requires only two -word memories due to the proposed in-place strategy. In addition, it uses one butterfly unit that can perform either one radix-4 butterfly or two radix-2 butterflies. The CFMR FFT processor using the 0.18 m SEC cell library consists of 37,000 gates excluding memories, requires only 640 clock cycles for a 512-point FFT and runs at 100 MHz. Therefore, the CFMR FFT processor can reduce hardware complexity and computation cycles compared with existing FFT processors.

Table 1: Comparison of the FFT architecture considering latency and area

Reference	Multiplier	Adder	MUX/ DMUX	Computati on Time
[10]	12	22	38	N/4 $\log_4 N$
[9]	9	19	38	N/4 $\log_2 N$
[8]	4	6	38	N/4 $\log_2 N$
[7]	4	6	N+12	N/4 $\log_2 N$

III. FFT ALGORITHM

A fast Fourier transform (FFT) is an algorithm to compute the discrete Fourier transform (DFT) and its inverse. Fourier analysis converts time (or space) to frequency and vice versa; an FFT rapidly computes such transformations by factorizing the DFT matrix into a product of sparse (mostly zero) factors. As a result, fast Fourier transforms are widely used for many applications in engineering, science, and mathematics. Show the butterfly operations for radix-2 DIF FFT in figure 2 and figure 3. The radix-2 algorithms are the simplest FFT butterfly algorithm.

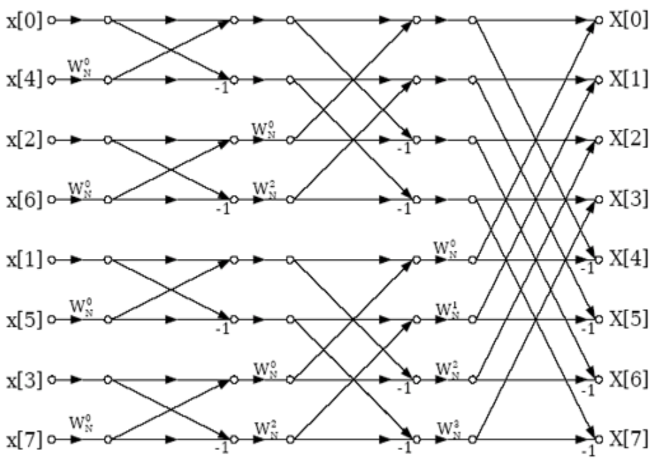

Figure 2: Radix-2 Decimation in Time Domain FFT Algorithm

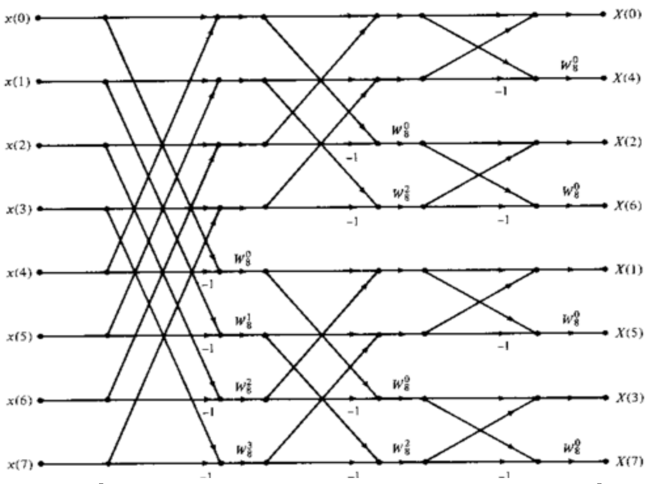

Figure 3: Radix-2 Decimation in Frequency Domain FFT Algorithm

IV. METHODOLOGY

Non programmable algorithm-specific processors can also be designed for the computation of FFT algorithms. The processors are designed mostly for fixed length FFTs. The architecture of an algorithmic-specific FFT processor is therefore optimized with respect to memory structure, control units, and processing elements. There are mainly three types of algorithm-specific processors: fully parallel FFT processors, column FFT processors, and pipelined FFT processors. All three types of algorithm-specific processors represent different mapping of the signal-flow graph for FFT to hardware structures. The hardware structure in a fully parallel FFT processor is an isomorphic mapping of the signal-flow graph.

To reduce the hardware complexity, a column or a pipelined FFT processor can be used. A set of process elements in a column FFT processor compute one stage at a time. The results are fed back to the same set of process elements to compute the next stage. For the long transform length, the routing for the processing elements is complex and difficult. For a pipelined FFT processor, each stage has its own set of processing elements. All the stages are computed as soon as data are available. Pipelined FFT processors have features like simplicity, modularity and high throughput. These features are important for real-time, in-place applications where the input data often arrive in a natural sequential order. We therefore select the pipeline architecture for our FFT processor implementation.

Radix-2 Multipath Delay Commutator:-

The Radix-2 Multipath Delay Commentator (R2MDC) architecture is the most straightforward approach to implement the radix-2 FFT algorithm using pipeline architecture. An 8-point R2MDC FFT is shown in Figure 3.

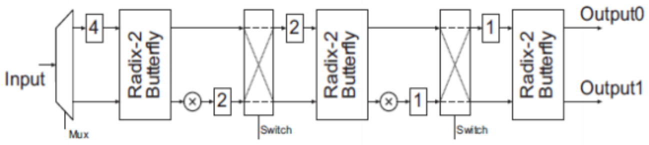

Figure 3: 8-point DIF R2MDC architecture.

When a new frame arrives, the first four input data are multiplexed to the top-left delay elements in the figure and the next four input data directly to the butterfly. In this way the first input data is delayed by four samples and arrives to the butterfly simultaneously with the fourth input sample. This completes the start-up of the first stage of the pipeline. The outputs from the first stage butterfly and the multiplier are then fed into the multipath delay commutator between stage 1 and stage 2. There are two paths (multipath) with delay elements and one switch (commutator). The multipath delay commutator alleviates the data dependency problem. The first and second outputs from the upper side of the butterfly are fed into the two upper delay elements. After this, the switch changes and the third and fourth outputs from the upper output of the first butterfly are sent directly to the butterfly at stage 2. However, the first and second outputs from the multiplier at the first stage are now delayed by the upper delay elements, which make the first and second outputs from the multiplier of the first stage arrive together with the fifth and sixth outputs from the top. The butterfly and the multiplier are idle half the time to wait for the new inputs. Hence the utilization of the butterfly and the multiplier is 50%. The total number of delay elements is 4 + 2 + 2 + 1 + 1 = 10 for the 8-point FFT. The total number of delay elements for an N-point FFT can be derived in similar way and is N/2+N/2+N/4+...+2, i.e., 3N/2-2. Each stage (except the last one) has one multiplier and the number of multipliers is $\log_2$ (N)–1.

V. CONCLUSION

The prime objective is to construct a FFT in order to have low power consumption and lesser area. The parameters (i) power consumption (ii) Area occupancy were given due consideration for comparing the proposed circuit with other FFTs. The circuits were simulated using Model-Sim 6.3c and synthesized with Xilinx ISE 14.1.The performance of various 64 point FFT such as Radix-2, Radix-4, split Radix, mixed -radix 4-2, R2MDC and the proposed modified R2MDC were carried out and their performance were analyzed with respect to the number of CLB slices, utilization factor and Power consumption.

REFERENCES

[1] Charles. Roth Jr., “Digital Systems Design using VHDL”, Thomson Brooks/Cole, 7th reprint, 2005.

[2] S. S. Kerur, Prakash Narchi, Jayashree C N, Harish M Kittur and Girish V A, “Implementation of Vedic multiplier for Digital Signal Processing”, International Conference on VLSI, Communication & Instrumentation (ICVCI) 2011, Proceedings published by International Joural of Computer Applications® (IJCA), pp.1-6.

[3] Himanshu Thapaliyal and M.B Srinivas, “VLSI Implementation of RSA Encryption System Using Ancient Indian Vedic Mathematics”, Center for VLSI and Embedded System Technologies, International Institute of Information Technology Hyderabad, India.

[4] Jagadguru Swami Sri Bharati Krishna Tirthaji Maharaja, “Vedic Mathematics: Sixteen simple Mathematical Formulae from the Veda”, Delhi (2011).

[5] Harpreet Singh Dhillon and Abhijit Mitra, “A Reduced-bit Multiplication Algorithm for Digital Arithmetic”, International Journal of Computational and Mathematical Sciences, Febrauary 2008, pp.64-69.

[6] Sumit Vaidya and Depak Dandekar. “Delay-power performance comparison of multipliers in VLSI circuit design”. International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4, July 2010.

[7] Pramod Kumar Mehe, Basant Kumar Mohanty, Sujit Kumar Patel, Soumya Ganguly, and Thambipillai Srikanthan, *“*Efficient VLSI Architecture for Decimation-in-Time Fast Fourier Transform of Real-Valued Data*”,* IEEE Transactions on Circuits And Systems—I: Regular Papers, Vol. 62, No. 12, December 2015.

[8] M. Ayinala, Y. Lao, and K. K. Parhi, “An in-place FFT architecture for real-valued signals,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 60, no. 10, pp. 652–656, Oct. 2013.

[9] Shashank Mittal, Md. Zafar Ali Khan and M.B. Srinivas, “Area Efficient High Speed Architecture of Bruun’s FFT for Software Defined Radio”, 1930-529X/07/$25.00 © 2007 IEEE.

[10] B. G. Jo and M. H. Sunwoo, “New continuous-flow mixed-radix (CFMR) FFT processor using novel in-place strategy,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 52, no. 5, pp. 911–919, May 2005.