# Design and Evaluation of an FFT Processor Utilizing Modified Sign Drift Graph with Novel Conflict-Free Deal with Schemes

A.S. SINDHU[1] ,M. SARITHA DEVI[2] ,
[1]*PG Student, Department of ECE, Godavari Institute of Engineering and Technology, Rajahmundry, Andhra Pradesh, India.*
[2]*Assistant Professor, Department of ECE, Godavari Institute of Engineering and Technology, Rajahmundry, Andhra Pradesh, India.*

***Abstract-*** In this transitority, we recommend a novel system which realizes conflict free method in memory-established FFT, of which the hardware complexity is simplified, considering the fact that only some extra registers are wanted and the control logic is equal in all levels. Additionally, we reward a modified signal drift graph to suit for the proposed conflict-free strategy. The modified signal go with the flow graph derives from the mixed-radix sign waft graph and has constant geometry property. Furthermore, continuous-waft is adopted to broaden the throughput. As a result, the proposed FFT processor has higher efficiency when compared with the prior reminiscence-based FFT processors. Simulation outcomes indicates that for the proposed eight to 2048-point FFT processor, the highest frequency is four hundred MHz by making use of a sixty-five-nm CMOS technology, and the discipline is zero.45 mm2 within the same

***Key terms-*** Quick Fourier transform (FFT), modified signal drift graph, steady geometry, clash-freestrategy, Memory-based,steady-drift

## I. INTRODUCTION

Speedy Fourier grow to be (FFT) is the main operation in digital signal processing, and is greatly used in digital verbal exchange, radar process and photo processing method, etc**.**Two kinds of commonly used FFT architectures are pipelined architecture [1, 2, 3, 4] and memory-based architecture [5, 6, 7, 8, 9]. The pipelined architecture achieves a higher throughput, but also inevitably with larger resource consumption and power consumption, since it uses independent processing element in each stage. The memory-based architecture has a lower throughput, but it has smaller resource consumption and power consumption, since it reuses the same processing elements in all stages. In this brief, we pay more attention to the memory-based architecture due to the consideration of the area. Reminiscence-established FFT consists of reminiscence, butterfly unit and corresponding control common sense. The memory is used to store the input data, the temporary data and the ultimate result. Since multiple data are written into one single memory simultaneously when processing the storage operation, there exists the problem of address conflict. As a way to solve the obstacle, clash-free process is proposed.

The conflict-free strategies proposed by [8] and [10] adopt multiplexers to change the output order in each stage. The storage order in each stage is different, thus the control logic varies in each stage. To realize the reordering, extra multiplexers are needed. Hence, it adds extra multiplexers and the control logic is relatively complicated. To optimize the conflict-free strategy further, we propose a novel method, using which the resource consumption is reduced compared with [10]. The proposed conflict-free approach is match for consistent geometry sign waft graph in [11, 12]. Besides, the property of constant geometry signal flow graph is that all stages are identical. By using the property, the logic becomes uniform which simplify the design.

In order to improve the performance of memory-based FFT, two methods are usually used. One way is to adopt high-radix butterfly unit, and the other is to adopt several parallel butterfly units. Of course, it is a tradeoff between the resource consumption and the performance. The proposed FFT processor adopts radix-4/2 mixed-radix algorithm to appreciate the tradeoff. The proposed mixed-radix sign waft graph is modified to receive the consistent geometry property. By way of making use of the property of the modified blended-radix signal drift graph, arbitrary 2n-factor FFT processor will also be carried out. In order to increase the throughput, two 2048-word memories are used to implement the continuous-flow computation.

## II. Modified signal flow graph for mixed-radix algorithm

In order to realize arbitrary 2n-point FFT with constant geometry property, the radix must equals to 2. Unfortunately, the performance is worse if the radix equals to 2. So we propose a modified radix-four/2 blended-radix signal flow graph which has regular geometry property to get higher performance. With the aid of utilizing the proposed sign flow graph, the proposed conflict-free strategy can be acquired.

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, 0 \le k \le N-1 \qquad (1)$$

Where

$$nk = (n_0 + 2 * \sum_{i=0}^{L-2} n_{i+1} 4^i)(\sum_{i=0}^{L-1} k_i 4^i)$$

Where L equals to the number of stage for the radix-4/2 mixed-radix FFT algorithmUsing the formula above, the signal flow graph of the radix-4/2 FFT and the corresponding twiddle factors in each stages can be obtained. The traditional radix-4/2 combined-radix sign go with the flow graph is shown in Fig. 1.

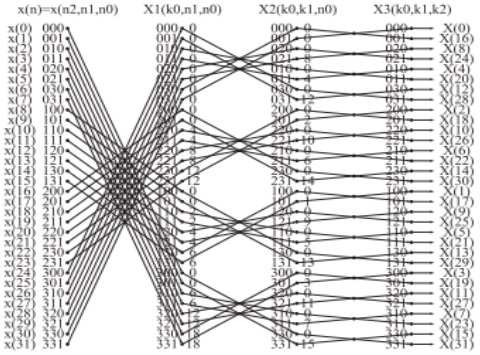


Fig.1: Traditional radix-4/2 combined-radix signal waft graph.

### A. Modified signal drift graph for combined-radix algorithm

According to the speculation of signal drift graph, it's identified that the sign go with the flow graphs before and after transformation are an identical as long as the relative operation position of each node keeps unchanged. Consequently, in an effort to acquire the modified signal go with the flow graph, we move the node and corresponding twiddle factors in Fig. 1. The output data in each stage are connected to the input node of the subsequent stage in sequence. Via using the approach mentioned, the modified sign glide graph is obtained which is proven in Fig. 2. Fig. 2 describes a 32-point FFT sign flow graph, and suggests that everyone phases are utterly identical. Utilizing this modified signal waft graph, the proposed clash-free procedure will also be realized.

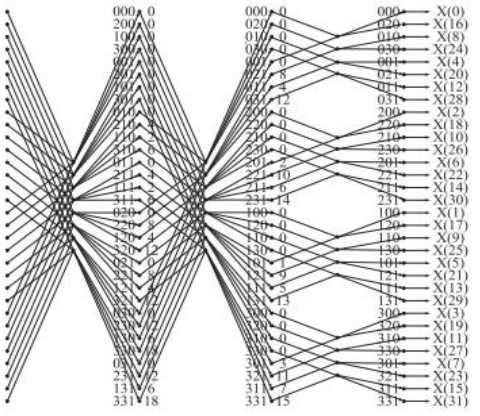

Fig.2: Modified radix-four/2 signal drift graph.

## III. DESIGN ISSUE OF THE FFT PROCESSOR

### A. Conflict-FreeStrategy

In this section, a novel conflict-free strategy is proposed. The method solves the address conflict by using the proposed signal flow graph which is illustrated in Fig. 2. The data flow chart in Fig. 3 corresponds to the modified sign waft graph in Fig. 2. In this 32-point FFT computation, a pair of memory is used to store input data, intermediate data and ultimate result. Each memory consists of four banks. The depth of each bank equals to 2. Each address in each bank stores four data. First, four input data are combined together one by one. Then the combined data are stored in Memory 1 in sequence. The storage sequence of the input data is shown in the first column of Fig. 3.

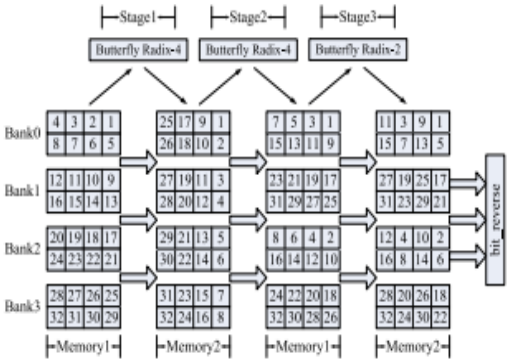


Fig.3: Conflict-free strategy for constant geometry approach.

After all the input data are stored, four data that follows the butterfly computation positions in the first stage of Fig. 2 are read out simultaneously. The four data in Memory 1 are stored in different banks, thus the reading address conflict does not exist anymore. For example, data 1, data 9, data 17 and data 25 in Memory 1 are read out concurrently at first, and data 2, data 10, data 18 and data 26 in Memory 1 are read out concurrently next time. Because each address stores four input data, we only extract one of them each time. After the four data in Memory 1 are read out simultaneou-sly, the fetched data are input to the shared butterfly unit simultaneously as shown in the Butterfly Radix-4 of Fig. 3. After the butterfly computation, 4 outcome of the butterfly computation are generated.

According to the input order of the 2nd stage of the modified signal float graph as shown in Fig. 2, 4 outcome are mixed together and stored in memory 2 in Fig. 3. For example, the butterfly results of data 1, information 9, data 17 and knowledge 25 in reminiscence 1 are combined together and saved within the first address of bank 0 in reminiscence 2. And the remainder storage sequence of the influence of butterfly computation in Stage 1 is shown in the 2nd column in Fig. 3.

In any case the operations above are finished, the first stage of FFT computation is completed. In view that the second stage of signal go with the flow graph is equal to the primary stage of signal go with the flow graph, the info waft chart in Stage 2 is identical to Stage 1. The computation in

Stage 3 is radix-2 computation. Thus, the data flow chart is a little bit different. According to the Stage 3 in Fig. 2, the computation sequence can be obtained. For example, the first data and the fifth data in Stage 3 operate radix-2 computation. The first data in Stage 3 which is stored in the first address in Bank 0 is data 1, as shown in the third column in Fig. 3. And the fifth data in Stage 3 which is stored in the second address in Bank 0 is data 9, as shown in the third column in Fig. 3. Data 1 and data 9 conduct radix-2 computation, and the results are stored following the sequence in the last stage of Fig. 2. Thus, they are stored in the first address of Bank 0 in Memory 2, as shown in the fourth column in Fig. 3.In order to get better performance, two pairs of radix-2 computation are conducted simultaneously. For example, data 1 and data 9, data 3 and data 11 in column 3 are computed in parallel. The results of them are combined as one data and stored in the first address in Bank 0 in Memory 2, as shown in the fourth column in Fig. 3. The storage sequence of the result of butterfly computation in Stage 3 is shown in the fourth column in Fig. 3. The address conflict is solved by using the proposed conflict-free strategy. The proposed method avoids the writing address conflict, since it writes the four data to one address instead of four. In addition, the proposed method avoids the reading address conflict, since it reads the four data from four different banks instead of one bank.

### B. Proposed FFT architecture

The proposed structure of the reminiscence-situated FFT processor based on the modified combined-radix signal go with the flow graph is proven in Fig. 4.By using the proposed architecture, the conflict-free strategy and the corresponding data flow chart mentioned in section A can be realized. To achieve the continuous-flow, increase the throughput and reduce the latency, the architecture contains two memories and each memory is comprised of four banks.The Register 1 in Fig. 4 is used as a shift register and it stores the input data one by one. After four input data are stored in the shift register, they are written into one address of the memory as a whole data. Before radix-4 butterfly computation, four input data of the butterfly unit must be output from the four banks of one memory.

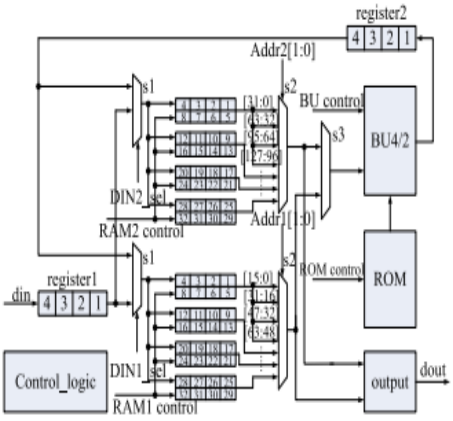


Fig.4: The architecture for the proposed 32-point FFT processor.

Each bank outputs a whole data which contains four data, the multiplexers in the second column are used to select the definite part of the whole data as the input data of the butterfly unit (BU4/2). The Register 2 in Fig. 4 outlets the four results of the BU4/2 at the same time. After the storage, the entire effect is written into the other memory. Before radix-2 butterfly computation, two pairs of input information of the butterfly unit have to be output from probably the most 4 banks of one memory. So as to fully grasp computation corresponding to the last stage in Fig. 2, the two whole data which contain four data are output from the definite bank one by one and input to the butterfly unit. s2 and s3 in Fig. 4 are used to control the data path. The control logic is used to generate the control signal and control the design to work in the desired way mentioned in Fig. 3. BU4/2 in Fig. 4 stands for butterfly unit and it can operate both radix-4 and radix-2 butterfly computation. ROM in Fig. 4 stores the twiddle factors and is also controlled by the control logic. The more detailed structure of BU4/2 is shown Fig. 5. When conducting radix4 computation, the four data are input directly to the basic BU4/2 module. Whereas, when conducting radix-2 computation, the two consecutive whole data which contains four data are stored in the Register 1 and Register 2 in Fig. 5. Then two pairs of data are selected from Register 1 and Register 2 through s3 and s4 and written into the basic BU4/2 module. The basic BU4/2 is shown in Fig. 6, and it reuses the resource of the radix-4 butterfly unit to implement two pairs of the radix-2 or radix-4 computation. The proposed normal BU4/2 is composed of three elaborate multipliers and eight elaborate adders. The multiplexers in Fig. 6 determine the data path and the radix selection concurrently

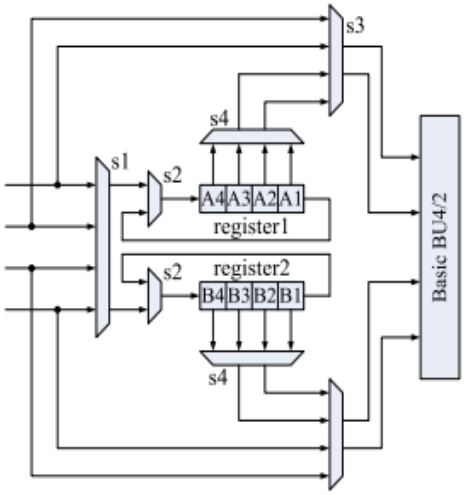


Fig.5: The architecture of radix-4/2 butterfly unit.

### A. Variable-point strategy

Based on the modified mixed-radix signal flow graph and the conflict-free strategy mentioned above, the configurable strategy can be obtained. For 8 to 2048-point configurable FFT, two groups of memories are needed. Each group of memory contains four memory banks whose depth equals to

128 and width equals to 128. When conducting configurable FFT, the maximum address in memory decreases as a result of the decrease of the FFT length. For example, when the point decreases to 1024 from 2048, the maximum address decreases to 64 from 128. Besides, the twiddle factor extraction follows the similar way.

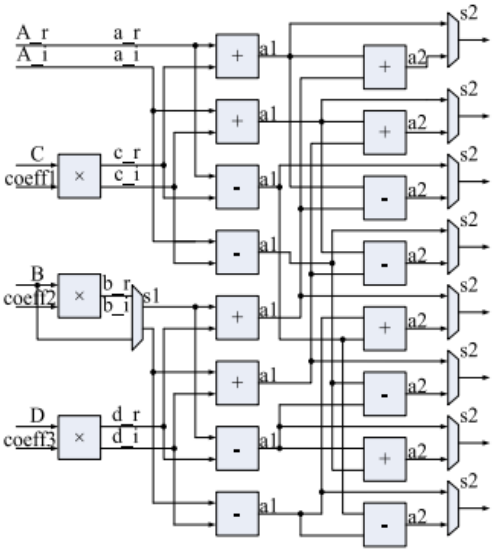


Fig.6: The architecture of basic radix-4/2 butterfly unit.

## IV. SIMULATION RESULTS

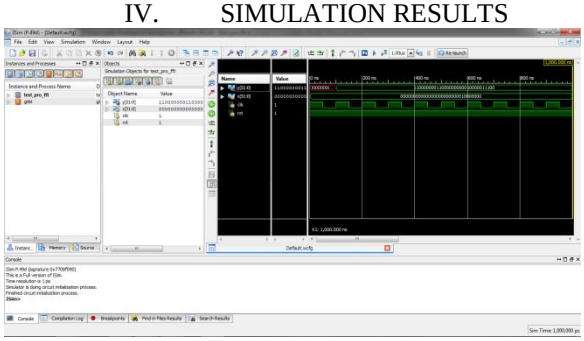

Fig.7: Simulation result of the proposed system

## V. CONCLUSION

In this paper, we propose an arbitrary 2n-point FFT processor by using a modified signal flow graph and corresponding novel conflict-free strategy, so that the logic control is simplified, the hardware resource is reduced, and the frequency is higher. Thus, our proposal is very suitable for real-time and lower-resource consumption system.

## VI. REFERENCES

[1]. J. Wang, et al.: "A mixed-decimation MDF architecture for radix-2k parallel FFT," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 24 (2016) 67 (DOI: 10.1109/TVLSI.2015.2402207).

[2]. Z. K. Wang, et al.: "A combined SDC-SDF structure for usual I/O pipelined radix-2 FFT," IEEE Trans. Very big Scale Integr. (VLSI) Syst. 23 (2015) 973 (DOI: 10.1109/TVLSI.2014.2319335).

[3]. Yu and M. H. Yen: "discipline-efficient 128- to 2048/1536-point pipeline FFT processor for LTE and mobile WiMAX techniques," IEEE Trans. Very big Scale Integr. (VLSI) Syst. 23 (2015) 1793 (DOI: 10.1109/TVLSI.2014. 2350017).

[4]. X. B. Yin, et al.: "useful resource-efficient pipelined architectures for radix-2 realvalued FFT with real datapaths," IEEE Trans. Circuits Syst. II, Exp. Briefs 63 (2016) 803 (DOI: 10.1109/TCSII.2016.2530862).

[5]. G. Jo and M. H. Sunwoo: "New steady-go with the flow combined-radix (CFMR) FFT processor utilizing novel in-situation strategy," IEEE Trans. Circuit Syst. I, Reg. Papers 52 (2005) 911 (DOI: 10.1109/TCSI.2005.846667).

[6]. P. Y. Tsai and C. Y. Lin: "A generalized conflict-free reminiscence addressing scheme for constant-glide parallel-processing FFT processors with rescheduling," IEEE Trans. Very giant Scale Integr. (VLSI) Syst. 19 (2011) 2290 (DOI: 10.1109/TVLSI.2010.2077314).

[7]. H. F. Luo, et al.: "effective reminiscence-addressing algorithms for FFT processor design," IEEE Trans. Very big Scale Integr. (VLSI) Syst. 23 (2015) 2162 (DOI: 10.1109/TVLSI.2014.2361209).

[8]. Z. G. Ma, et al.: "A novel reminiscence-founded FFT architecture for precise-valued symptoms based on a radix-2 decimation-in-frequency algorithm," IEEE Trans. Circuits Syst. II, Exp. Briefs sixty two (2015) 876 (DOI: 10.1109/TCSII.2015. 2435522).

[9]. M. Garrido, et al.: "A 4096-factor radix-four memory-founded FFT making use of DSP slices," IEEE Trans. Very massive Scale Integr. (VLSI) Syst. 25 (2017) 375 (DOI: 10.1109/TVLSI.2016.2567784).

[10]. K. F. Xia, et al.: "A memory-based FFT processor design with generalized efficient conflict-free address schemes," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 25 (2017)1919(DOI:10.1109/TVLSI.2017.2666820).

[11]. J. H. Takala, et al.: "Conflict-free parallel memory access scheme for FFT processors," Proc. IEEE ISCAS (2003) 524 (DOI: 10.1109/ISCAS.2003. 1205957).

[12]. J. Kwong and M. Goel, "A high performance split-radix FFT with constant geometry architecture," Proc. IEEE DATE (2012) 1537 (DOI: 10.1109/DATE. 2012.6176717).