Trends in Engineering and Technology (NCTET-2K17)
International Journal of Advanced Engineering Research and Science (IJAERS) *Special Issue-4*
https://dx.doi.org/10.22161/ijaers/nctet.2017.ece.20 *ISSN: 2349-6495(P) | 2456-1908(O)*

Modified Butterfly Structure for Split Radix FFT Processors

K.Soundharya[1], P.R Sidda Reddy[2]
[1]M.Tech Scholar, Associate Professor [2]
[1, 2] Department of ECE, SVCE, Tirupati, India
Email: [1] soundharya.kalathuru@gmail.com , siddu.polimera@gmail.com [2] ,

Abstract: *Split-radix quick Fourier transform (SRFFT) is a perfect possibility for the execution of a low-power control FFT processor, since it has the most reduced number of math operations among all the FFT calculations. In the outline of such processors, a proficient tending to scheme for FFT information and also twiddle factors is required. The signal flow graph diagram of SRFFT is the same as radix-2 FFT, and consequently, the conventional address generation plans of FFT information could likewise be connected to SRFFT. Be that as it may, SRFFT has unpredictable areas of twiddle factors and denies the utilization of radix-2 address generation strategies. This short displays a shared-memory low-control SRFFT processor architecture. We demonstrate that SRFFT can be computed by utilizing modified radix-2 butterfly unit. The butterfly unit abuses the multiplier-gating system to save dynamic power to the detriment of utilizing more equipment assets. Moreover, two novel address generation algorithms for both the trivial and nontrivial twiddle factors are developed. simulation comes about demonstrate that contrasted and the ordinary radix-2 shared-memory executions, the proposed configuration accomplishes more than 20% lower control utilization when computing a 1024-point complex-value transform.*

INTRODUCTION

The fast Fourier transform (FFT) is a standout amongst the most essential and principal calculations in the advanced digital signal processing area. Since the disclosure of FFT, numerous variations of the FFT calculation have been produced, for example, radix-2 and radix-4 FFT. In 1984, Duhamel and Hollmann [1] proposed another variation of FFT calculation called split-radix FFT (SRFFT). Their calculation requires minimal number of multiplications and additions among all the known FFT algorithms. Since arithmetic operations significantly contribute to overall system power consumption, SRFFT is a good candidate for the implementation of a low-power FFT processor.

Existing System:

In general, all the FFT processors can be ordered into two main groups: pipelined processors or shared-memory processors. Cases of pipelined FFT processors can be found in [2] and [3]. A pipelined architecture gives high throughputs, however it requires more equipment assets in the meantime. One or various pipelines are frequently actualized, each comprising of butterfly units and control logic.
Conversely, the shared memory-based architecture requires minimal measure of hardware resources to the expense of slower throughput. In the radix-2 shared-memory architecture, the FFT data are sorted out into two memory banks. At each clock cycle, two FFT information are given by memory banks and one butterfly unit is utilized to process the data.
At the following clock cycle, the computation comes about are composed back to the memory banks and replace the old data. The extent of this brief is constrained to the shared memory architecture. In the shared-memory architecture, an efficient addressing scheme for FFT data as well as coefficients (called twiddle factors) is required For the fixed radix FFT.
For split-radix FFT, it conventionally includes a L-shaped butterfly data path whose unpredictable shape has uneven latencies and makes scheduling troublesome. In this brief, we demonstrate that the SRFFT can be computed by utilizing an modified radix-2 butterfly structure. Our commitment comprises of mapping the split-radix FFT algorithm to the shared memory architecture, utilizing the lower multiplicative multifaceted nature of the algorithm to reduce the dynamic power and developing two novel twiddle factor addressing schemes for the split-radix FFT.

https://dx.doi.org/10.22161/ijaers/nctet.2017.ece.20 *ISSN: 2349-6495(P) | 2456-1908(O)*

Assume that we have $N=2^s$ point FFT radix-2 FFT require S passes to finish the computation, as shown in Fig1.

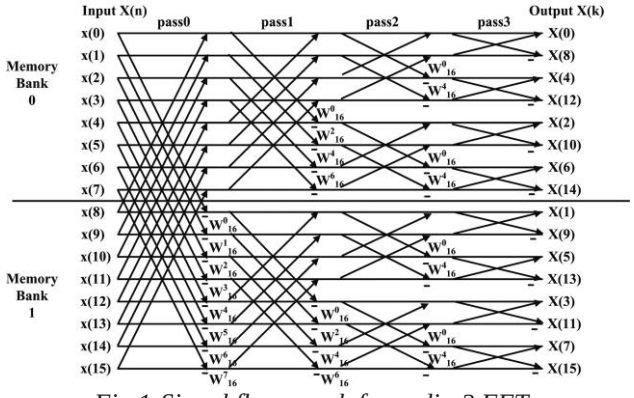


Fig 1:Signal flow graph for radix-2 FFT.

Disadvantages

- Dynamic Power consumption is high.
- Number of multiplications is high.

I

II. PROPOSED SYSTEM.

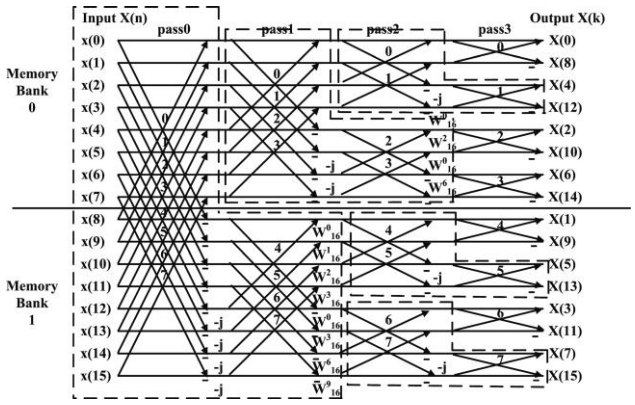


Fig 2: Signal flow graph for SRFFT.

Every L butterfly contains two nontrivial complex multiplications,

also, in this manner, the total number of nontrivial complex multiplications MSR in SRFFT is

$$M\text{SR} = [(3S - 2)2^{S-1} + (-1)^S]2/9.$$

In the $(S - 1)$th pass, the number of SR butterfly N^{S-1} is

$$N^{S-1} = [2 + (-1/2)^{S-2}]N/12 .$$

However, in the $(S - 1)$th pass, each L butterfly does not contain any nontrivial twiddle factors and hence, the total number of nontrivial multiplications M_SR in SRFFT is

$$M\text{\_SR} = M\text{SR} - 2^{NS-1}.$$

For the conventional radix-2 FFT, the total number of complex multiplications $MR2$ is

$$MR2 = 2^{S-1}(S - 1).$$

A. Shared-Memory Architecture

The architecture of shared-memory processor is appeared in Fig. 3. The FFT data and the twiddle factors are stored in the RAM and ROM banks, separately. We watched that the flow graph of split-radix algorithm is the same as radix-2 FFT with the exception of the locations and estimation values of the twiddle factors and hence, the conventional radix-2 FFT data address generation schemes could likewise be connected to SRFFT (RAM address generator). however, the mixed radix property of SRFFT algorithm prompts the unpredictable areas of twiddle factors and disallows any conventional address generation algorithm (ROM address generator).

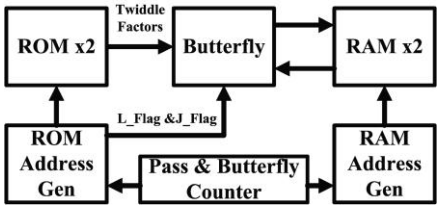


Fig 3: Shared-memory architecture.

B. Modified Radix-2 Butterfly Unit

we proposed an modified butterfly unit which is appeared in Fig. 4. The structure of this butterfly unit is controlled by the way that the SRFFT has multiplications of both the upper and lower legs. To anticipate superfluous exchanging action, we put the check gating registers in the multiplier way and a couple of registers are set at the address port of memory banks to synchronize the entire design. The way to utilize this architecture is the knowing about which butterflies require no multiplications (the complex multipliers are then skipped), trivial multiplications (swapping), and nontrivial multiplications (utilizing complex multipliers).

https://dx.doi.org/10.22161/ijaers/nctet.2017.ece.20 *ISSN: 2349-6495(P) | 2456-1908(O)*

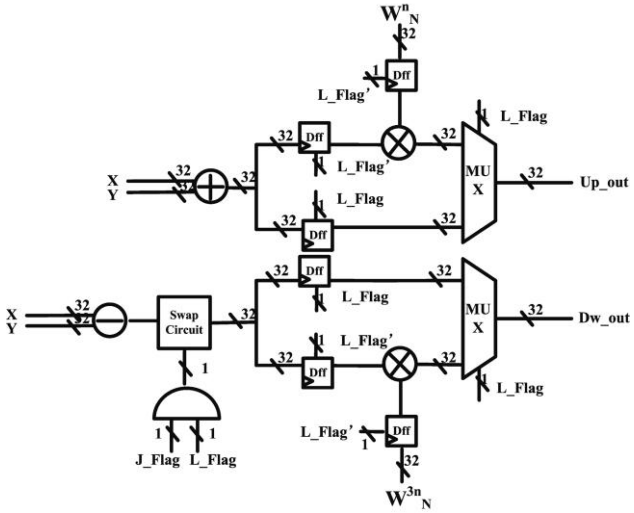


Fig 4: Modified butterfly structure.

C. Address Generation of Twiddle Factors

The flow graph for the 16-point SRFFT is appeared in Fig. 2. In Fig. 2, there are two kinds of twiddle factors: j and Wn. For those multiplications including j is called trivial multiplications, on the grounds that these operations are basically the swapping of the real and imaginary part of the multiplier, henceforth no multiplication is included. For those multiplications including Wn are called nontrivial multiplications, since complex multipliers are utilized to finish these operations. In Fig. 2, every area encompassed by the dashed lines is called one L block which is formed by L butterflies in each pass [8]and there are absolutely five L blocks for a 16-point SRFFT.

I

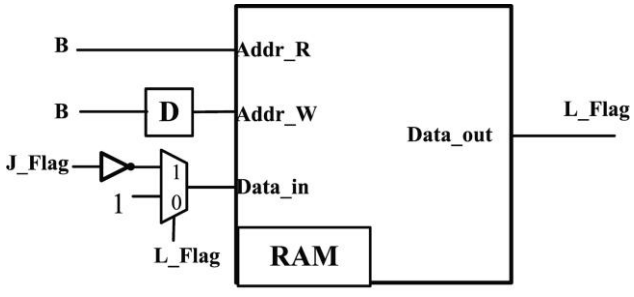


Fig 5: L_Flag structure.

L_Flags could be efficiently actualized as a RAM block,as appeared in Fig. 5. Butterfly counter B is filled in as the read address of RAM and at each clock cycle, the relating L_Flag esteem for the present butterfly is given. The udated L_Flag esteem for the following pass is composed to this memory at next clock cycle. The measure of this memory is 2S−1 bits, which equivalents to the quantity of butterflies in each pass. Such a size is trivial on the present day field-programmable gate array exhibit (FPGA). For instance, for a 2048-point FFT just 1 kbit is required. J_Flag is a combinational signal. The estimation of this variable relies upon the butterfly counter B. In the Pth pass, J_Flag equivalents to

$$bS-2-P.$$

In the last pass, L_Flag is set to one and J_Flag is set to zero. At the point when nontrivial multiplication is required, twiddle factors should be recovered from the ROM banks. Not at all like traditional conventional that stores all the Wn in one ROM bank, we sort out Wn in two ROM banks: one stores Wn for the upper leg of the butterfly unit and alternate stores those for the lower leg of the butterfly unit. the substance of the two ROM banks for a 16-point SRFFT. Begun inpass 1, in the Pth pass the address of every ROM bank is given by $bS-2-PbS-3-P \ldots b00 \ldots 0$ (following $(P-1)$ '0' s).

It merits specifying that in regular usage, the twiddle factors are required for each butterfly so ROM banks are constantly empowered, and in our execution, the L-Flag flag can be utilized as the empower motion for the ROM banks, since that if the butterfly has a place with the L block, no multiplication is required. This could prompt further power reduction. Both the address generation of FFT data and twiddle factors relies upon certain butterfly preparing request in each pass. Other than Xiao's [6] techniques (Fig. 2), there are different techniques for requesting the butterfly sequence, for example, [5]. We have additionally built up the address generation strategies for this sort of butterfly sequence utilizing the comparable thoughts expressed previously. The subtle elements are not talked about here and we just give the conclusions. In each go aside from the last one, J-Flag equivalents to

$$bS-2.$$

The address for each ROM bank is given by

$$bS-2bS-3 \ldots b1.$$

Advantages:

- Dynamic power consumption is decreased.

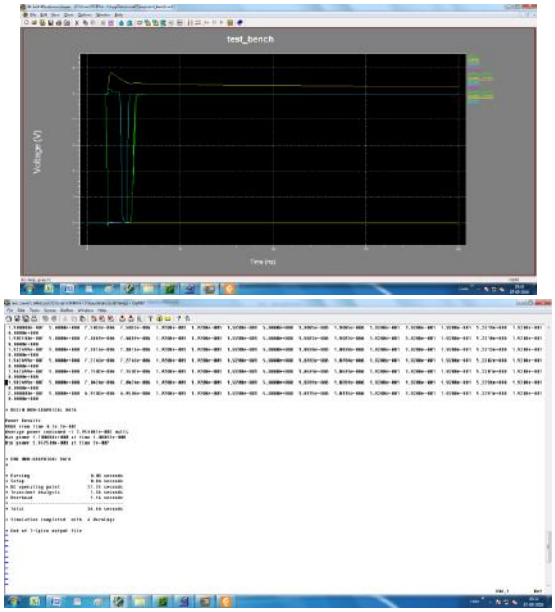
https://dx.doi.org/10.22161/ijaers/nctet.2017.ece.20 *ISSN: 2349-6495(P) | 2456-1908(O)*

- Number of multiplication is decreased.

Software implementation:

- Xilinx ISE 14.2

IV SIMULATION RESULTS



V CONCLUSION

In this short, a shared- memory-based SRFFT processor is proposed. The proposed technique lessens the dynamic power consumption to the detriment of more hardware resources. We likewise display two tending to schemes for both the trivial and nontrivial twiddle factors. Since SRFFT has the base number of multiplications contrasted and different types of FFT, the outcomes could be more ideal in the sense of floating point operations.

REFERENCES

[1] P. Duhamel and H. Hollmann, "Split radix' FFT calculation," Electron. Lett., vol. 20, no. 1, pp. 14–16, Jan. 1984.

[2] M. A. Richards, "On hardware implementation of the split-radix FFT," IEEE Trans. Acoust., Speech Signal Process., vol. 36, no. 10, pp. 1575–1581, Oct. 1988.

[3] J. Chen, J. Hu, S. Lee, and G. E. Sobelman, "hardware efficient radix-25/16/9 FFT for LTE systems," IEEE Trans. very Large Scale Integr. (VLSI) Syst., vol. 23, no. 2, pp. 221–229, Feb. 2015.