Power Reduction in SR FFT Processor Using Multi Bit Flip Flop Method

[1]P. Syamala Devi, [2]Dr.K.Murali Babu

[1]Assistant Professor,[2]Associate Professor

[1]ECE Department

[1]Annamcharya Institute Of Technology and Sciences,Rajampet,India

Abstract: Fast fourier transform (FFT) is one of the most important and fundamental algorithm in digital signal processing area. Split radix fast fourier transform (SRFFT) algorithm requires the least number of multiplications and additions among all the known FFT algorithms, which contribute to overall system power consumption. In the design of such processors, modified radix-2 Shared memory architecture can be used instead of Pipelined FFT architecture. In contrast Shared memory based architecture requires least amount of hardware resources at the expense slower throughput. To implement this Shared memory architecture, **Multi-Bit FlipFlop** (MBFF) and **Super fast low power SRAM** (SFLP SRAM) cell are required. The SRFFT can be computed by using a modified radix-2 butterfly unit. The butterfly unit exploits the multiplier gating technique to save dynamic power. In addition, two address generation algorithm are developed for both real and imaginary parts of twiddle factors. In simulation results the power is measured by T-Spice using Tanner tool. Hence the proposed architecture will save more power when it comes to larger points of FFT.

Keywords: Address generation, low power, radix-2, split-radix fast Fourier transform (SRFFT), MBFF, SFLP SRAM, Twiddle factors.

I. INTRODUCTION

The fast Fourier transform (FFT) is an efficient algorithm that computes the discrete Fourier transform (DFT) of a sequence, or its inverse. Which reduces the number of computations. Since the discovery of FFT, many variants of the FFT algorithm have been developed, such as radix-2 and radix-4 FFT. A new variant of FFT called Split-Radix FFT (SRFFT) was developed by Duhamel and Hollman in the year 1984. Their algorithm requires the least number of multiplications and additions among all the FFT algorithms. Since arithmetic operations significantly contribute to overall system power consumption. The idea behind their algorithm is the application of a radix-2 index mapping to the even-index terms and a radix-4 mapping to the odd-index terms, which results in an "L" shaped butterfly. The signal flow graph of split-radix FFT is the same as radix-2 FFT except for the location and value of twiddle factors. Therefore address generation scheme for conventional radix-2 FFT algorithm could also be applied to SRFFT.

SRFFT is a good candidate for the implementation of a low-power FFT processor. In general, all the FFT processors can be categorized into two main groups: pipelined processors or shared-memory processors. A pipelined architecture provides high throughputs, but it requires more hardware resources at the same time. One or multiple pipelines are often implemented, each consisting of butterfly units and control logic. In contrast, the shared-memory-based architecture requires the least amount of hardware resources at the expense of slower throughput. In the radix-2 shared-memory architecture, the FFT data are organized into two memory banks. At each clock cycle, two FFT data are provided by memory banks and one butterfly unit is used to process the data. At the next clock cycle, the calculation results are written back to the memory banks and replace the old data. The scope of this brief is limited to the shared-memory architecture.

The FFT calculations are arranged into two general classes, to be specific, the decimation in-time (DIT) and the decimation in frequency (DIF) uncommonness calculations. The key contrasts between the two are appeared in Fig. If there should arise an occurrence of DIF calculation (Fig. (a)), the info tests are bolstered to the registering structure in their regular request, while the yield is created in bit-switched request. Then again, if there should be an occurrence of DIT calculation (Fig. (b)), the info tests need bit-inversion reordering before being handled, while the yield FFT coefficients are created in characteristic request.

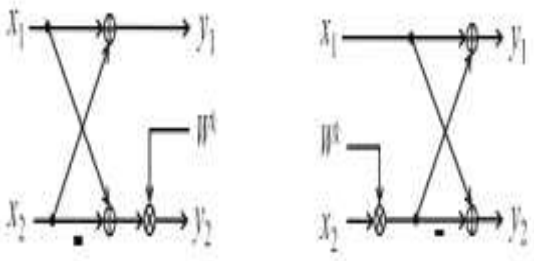

Figure (a) DIF FFT butterfly (b) DIT FFT butterfly

II.RELATED WORK

FFT ALGORITHM

The conventional signal and image processing applications requires high computational power based on Fast Fourier Transform (FFT) in addition to the ability to choose the algorithm and architecture. The Fast Fourier Transform (FFT) converts a time based signal into its corresponding frequency based signal by manipulating the sum of orthogonal components. The time based signal spectrum is indicated by the phase and amplitude of those components. Inverse Fast Fourier Transform (IFFT) does the reverse process, thus converting the spectrum back to time signal. Every point of data present in the spectrum is called a bin.

RADIX 2 FFT

When 'N' is a power of 2, say $N=2^K$, where K>1 is an integer, then the above DIT decomposition can be performed K-1 times, until each DFT is length 2. A length 2 DFT requires no multiplies. The overall result is called a radix 2 FFT. A different radix 2 FFT is derived by performing decimation in frequency.

A split radix FFT is theoretically more efficient than a conventional radix-2 algorithm because it minimizes real arithmetic operations. The term `"split radix" refers to a DIT decomposition that combines portions of one radix 2 and two radix 4 FFTs. On modern general-purpose processors, however, computation time is often not minimized by minimizing the arithmetic operation count.

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

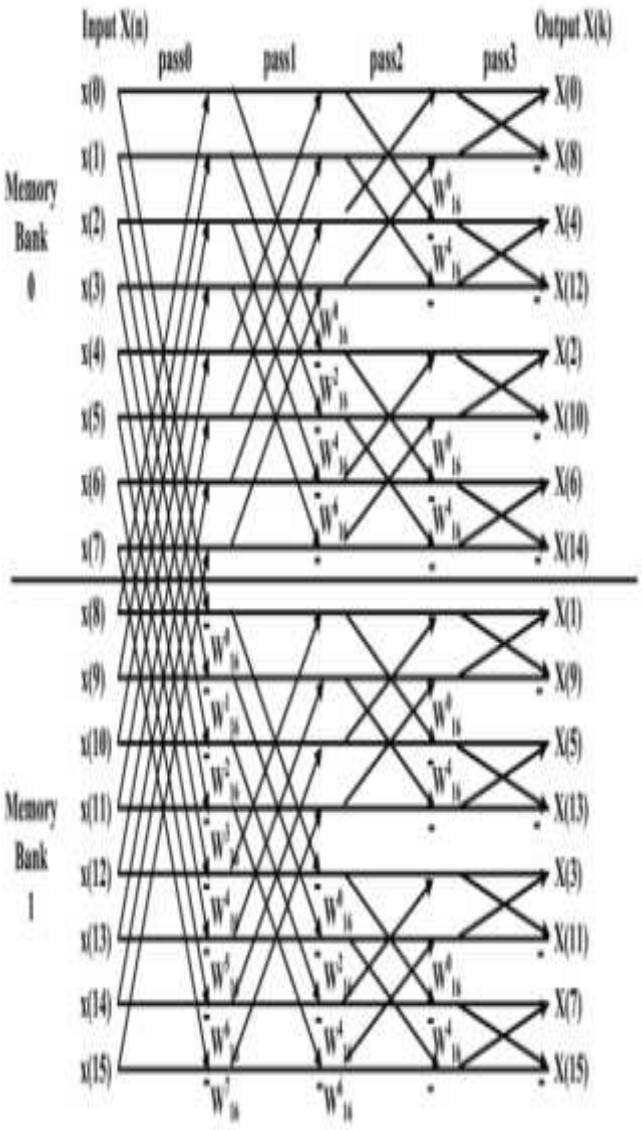

Fig 1 Signal flow graph for radix-2 FFT

III.PROPOSED METHOD

For split-radix FFT, it conventionally involves an L-shaped butterfly data path whose irregular shape has uneven latencies and makes scheduling difficult. In this brief, we show that the SRFFT can be computed by using a modified radix-2 butterfly structure. This butterfly structure uses the multi-Bit FlipFlop (MBFF) architecture in place of D-Flipflop (Single-Bit Flipflop) for increasing the speed of the architecture. The signal flow graph of proposed Split-radix FFT is shown in Fig 2.

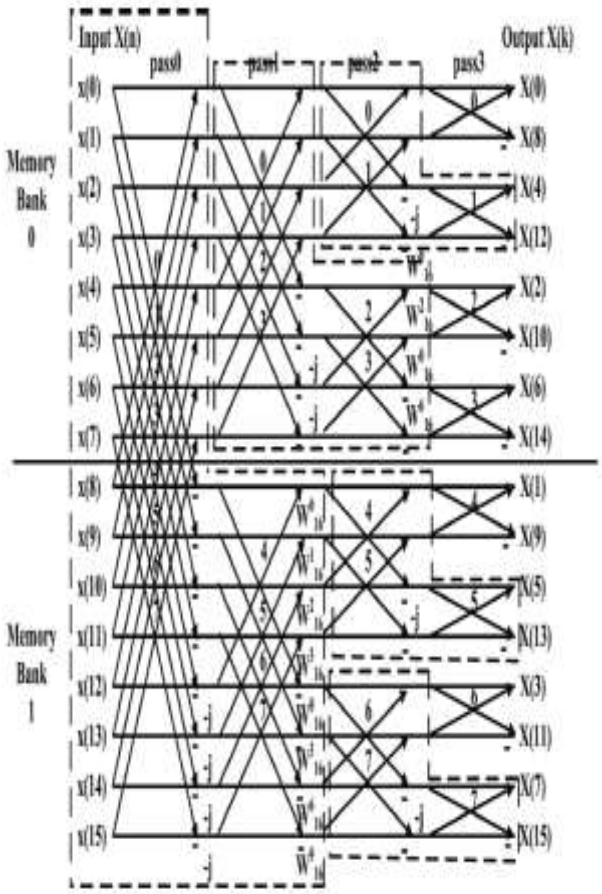

Fig 2. Signal flow graph for proposed SRFFT

A) Shared-Memory Architecture

The architecture of shared-memory processor is shown in Fig3. The FFT data and the twiddle factors are stored in the RAM and ROM banks, respectively. We observed that the flow graph of splitradix algorithm is the same as radix-2 FFT except for the locations and values of the twiddle factors and therefore, the conventional radix-2 FFT data address generation schemes could also be applied to SRFFT (RAM address generator). However, the mixed-radix property of SRFFT algorithm leads to the irregular locations of twiddle factors and forbids any conventional address generation algorithm (ROM address generator).

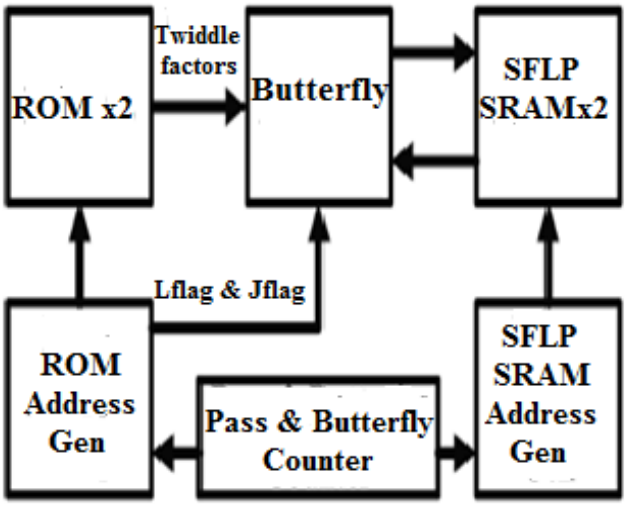

Fig 3.Shared-memory architecture.

A) Modified Radix-2 Butterfly Unit

In our previous work we proposed a modified butterfly unit which is shown in Fig. 4. The structure of this butterfly unit is determined by the fact that the SRFFT has multiplications of both the upper and lower legs. To prevent unnecessary switching activity, we put the clock gating registers in the multiplier path and a few registers are placed at the address port of memory banks to synchronize the whole design. The key to use this architecture is the knowing about which butterflies require no multiplications (the complex multipliers are then skipped), trivial multiplications (swapping), and nontrivial multiplications (using complex multipliers).

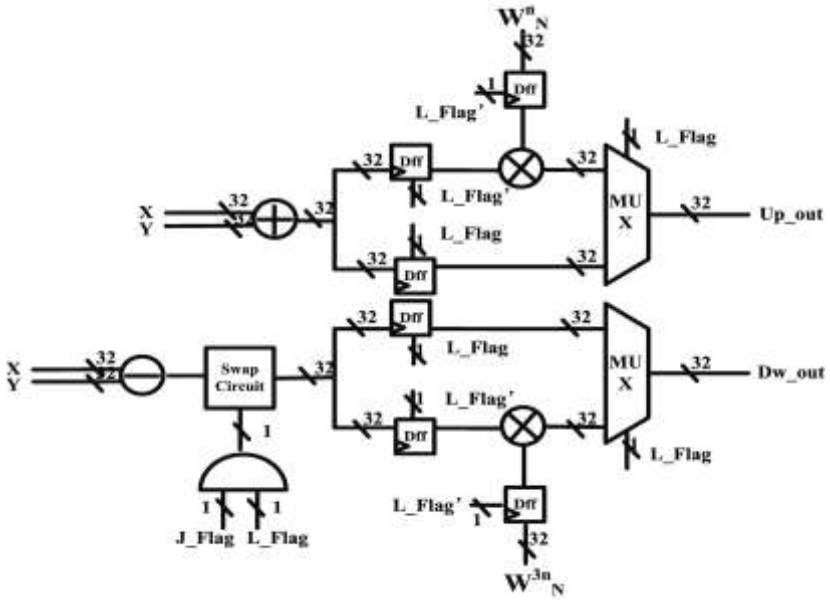

Fig 4.Modified butterfly structure

B)SFLP SRAM

The complete schematic of the proposed Super-Fast Low-Power (SFLP) SRAM cell is shown in Fig5. The SFLP SRAM cell is divided into two circuits: write circuit and read circuit. The write circuit is similar to the 8T cell and it is used to perform the write operation. The read operation is performed by two transistors (N7 and N5) which are used to perform the read operation. Transistor N7 is used to decouple the cell node from the read bit line during write operation/sleep mode.

During write operation read word line (RWL) is set to low so that write circuit completely decouples from the read circuit. The write operation in the proposed cell is performed by setting the bitlines (BL and BLB) to the desired logic. During write operation, the proposed SRAM cell behaves as the 8T cell. In the SFLP cell, the switching behaviors of tail transistors are controlled by the respective logic on the storage nodes Q and nQ.

A read operation (RWL = 1, WL = 0) is performed by reading the data with help of the transistors N7 and N5. In read 1 operation, the transistor N5 is turned OFF, flips the node S to logic high, without allowing RBL to discharge which results lower power consumption. When node nQ stores data “1” (read 0 operation), transistor N5 turns ON. Now, the bitline (RBL) discharges through read access transistor N7 and pull down transistor N5. Since, storage nodes Q and nQ are completely isolated from bitlines during read operation; the voltage of the node which stores 0 is strictly maintained at the ground level.

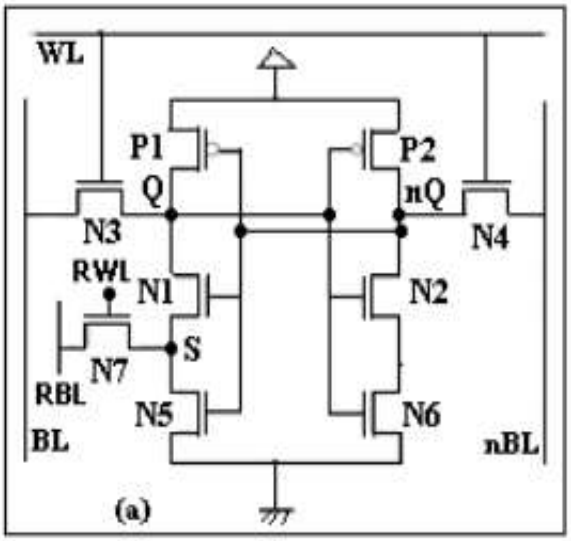

Fig5: SFLP SRAM Cell

A) Address Generation of Twiddle Factors

The flow graph for the 16-point SRFFT is shown in Fig. In Fig. 2, there are two kinds of twiddle factors: j and Wn. For those multiplications involving j is called trivial multiplications, because these operations are essentially the swapping of the real and imaginary part of the multiplier, hence no multiplication is involved. For those multiplications involving Wn are called nontrivialmultiplications, because complex multipliers are used to complete these operations. In Fig. 2, each area surrounded by the dashed lines is called one L block which is formed by L butterflies in each pass and there are totally five L blocks for a 16-point SRFFT.

```
1: Initialization: set all the L_Flag to 1
2: for P = 0 to S − 1 do
3:   for B = 0 to 2^(S−1) do
4:     J_Flag ← b_(S−2−P)
5:     if L_Flag_b = 1 then
6:       if J_Flag = 1 then
7:         Multiply trivial twiddle factor j (swapping)
8:         ROM_Address ← don'tcare
9:         L_Flag_b ← 0
10:      else
11:        No Multiplication Required
12:        ROM_Address ← don'tcare
13:        L_Flag_b ← 1
14:      end if
15:    else
16:      Multiply non-trivial twiddle factor W_n
17:      L_Flag_b ← 1
18:      ROM_Address
19:      ← b_(S−2−P) b_(S−3−P) ... b_0 0...0
20:    end if
21:  end for
22: end for
```

Pseudocode for tracking trivial and nontrivial twiddle factors

III.RESULTS

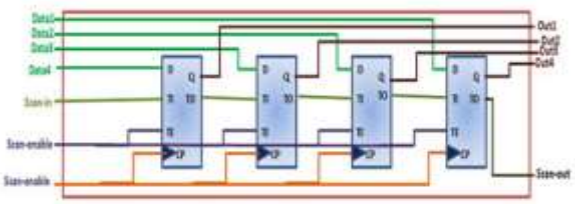

Fig: Schematic Circuit for Multi-Bit FlipFlop

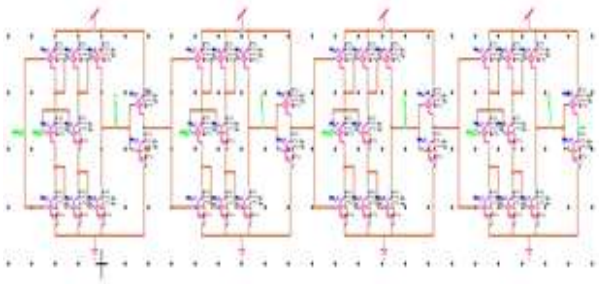

Fig: Simulation Circuit for Multi-Bit FlipFlop

Power consumption is an important issue in modern high frequency and low power designs. In this proposed method Multi-Bit FlipFlop (MBFF) is used to reduce the clock system power. The scaling with multiple supply voltage is an efficient way to minimize the dynamic power consumption.

Block diagram

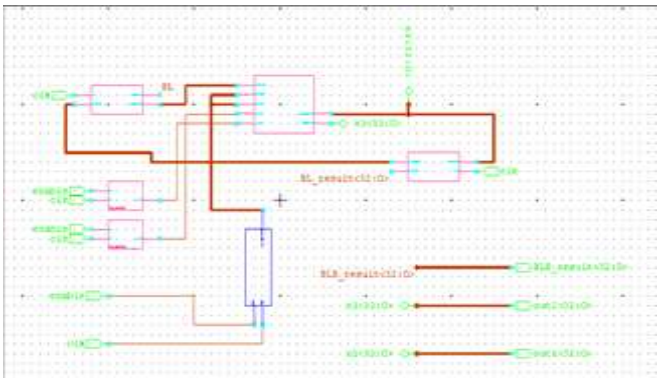

Fig: Simulation Block diagram of Shared memory based SRFFT processor

Power Results:

```
VVS_1 from time 0 to 1e-006
Average power consumed -> 5.396726e-001 watts
Max power 2.313182e+000 at time 9.90694e-007
Min power 1.818677e-001 at time 3.00313e-008

VVS_2 from time 0 to 1e-006
Average power consumed -> 1.873827e-007 watts
Max power 1.795198e-004 at time 1.00687e-007
Min power 0.000000e+000 at time 0

VVS_6 from time 0 to 1e-006
Average power consumed -> 3.665827e-011 watts
Max power 7.800072e-006 at time 1.04539e-008
Min power 0.000000e+000 at time 0
```

Delay Report:

```
tdelay = 0.0000e+000
   Trigger = 4.0900e-008
   Target  = 4.0900e-008

tdelay1 = 0.0000e+000
   Trigger = 4.0900e-008
   Target  = 4.0900e-008
```

Simulation results:

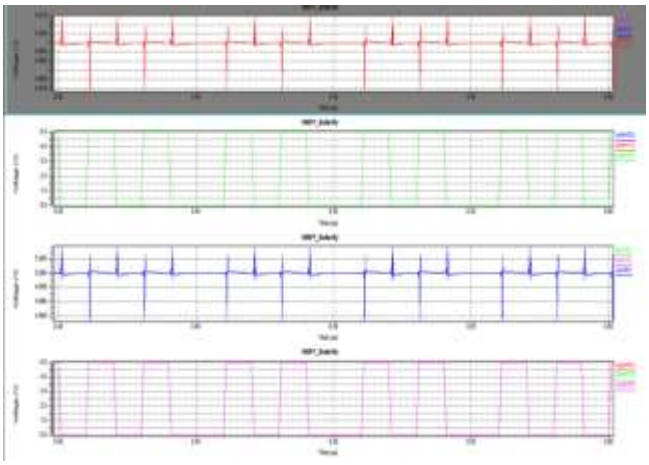

Fig: Simulation Results for Outputs corresponding inputs

In this paper, the proposed DIF SRFFT having normal inputs and produced corresponding Bit-Reversal Outputs.

I. CONCLUSION

In this paper, design a low power Shared memory SRFFT processor structure using Tanner Tool 13.0. The proposed design used Multi Bit Flip Flop (MBFF) and SFLP SRAM for reduces the dynamic power consumption at the expense of more hardware resources. Because the SRFFT is known to have lower multiplicative complexity than the both conventional radix-2 and radix-4 algorithms. Thus, it does not only achieve the minimum hardware requirement but also saves the power and increases the maximum clock rate at the same time. The Proposed design achieves over 30% lower power consumption when computing a 1024-point complex-valued transform. In future, it may save significant power for OFDM systems with long length FFT.

REFERENCES

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

[2] M. A. Richards, "On hardware implementation of the split-radixFFT," 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 mixedradix-25/16/9 FFT for LTE systems," IEEE Trans. Very Large ScaleIntegr. (VLSI) Syst., vol. 23, no. 2, pp. 221–229, Feb. 2015.

[4] L. G. Johnson, "Conflict free memory addressing for dedicated FFThardware," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.,vol. 39, no. 5, pp. 312–316, May 1992.

[5] D. Cohen, "Simplified control of FFT hardware," IEEE Trans. Acoust.,Speech, Signal Process., vol. 24, no. 6, pp. 577–579, Dec. 1976.

[6] X. Xiao, E. Oruklu, and J. Saniie, "An efficient FFT engine with reducedaddressing logic," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 55,no. 11, pp. 1149–1153, Nov. 2008.

[7] Z. Qian, N. Nasiri, O. Segal, and M. Margala, "FPGA implementationof low-power split-radix FFT processors," in Proc. 24th Int. Conf. Field

Program. Logic Appl., Munich, Germany, Sep. 2014, pp. 1–2.

[8] A. N. Skodras and A. G. Constantinides, "Efficient computation of thesplit-radix FFT," IEE Proc. F-Radar Signal Process., vol. 139, no. 1,pp. 56–60, Feb. 1992.

[9] H. V. Sorensen, M. T. Heideman, and C. S. Burrus, "On computing the split-radix FFT," IEEE Trans. Acoust., Speech Signal Process., vol. 34,no. 1, pp. 152–156, Feb. 1986.

[10] J. Kwong and M. Goel, "A high performance split-radix FFT withconstant geometry architecture," in Proc. Design, Autom. Test Eur. Conf.Exhibit. (DATE), Dresden, Germany, Mar. 2012, pp. 1537–1542.

[11] W.-C. Yeh and C.-W. Jen, "High-speed and low-power split-radixFFT," IEEE Trans. Signal Process., vol. 51, no. 3, pp. 864–874,Mar. 2003.