Radix-2 Pipelined FFT Processor with Gauss Complex Multiplication Method and Vedic Multiplier

Vamshipriya. Bogireddy
School of Electronics Engineering(SENSE)
Vit university,chennai

P. Augusta Sophy
School of Electronics Engineering(SENSE)
Vit university,Chennai

***Abstract*—Optimizing power consumption is an important design goal for digital signal processing systems (DSPs), but it is often difficult to achieve. The principal reasons for this are maximum operating temperature and, for portable applications, batterylife. Because of the relatively greater complexity, the power dissipation in digital signal processing (DSP) applications is of special significance, and low power design technique, are now emerging. This paper focuses on designing low power consuming and high speed FFT processor. In this paper an efficient 16-point DIF-FFT processor is designed by employing different low power techniques at various levels of design abstraction i.e, pipelining method at architectural level, Gauss complex multiplication method and Vedic multiplier at algorithmic level. The Vedic multiplier plays a major role in this design because due to its different approach low power consumption as well as high speed is possible .**

Keywords –Vedicmultiplier, Gaussmultiplication method, Low power consumption, pipelined architecture, In-place addressing, Decimation in frequency(DIF).

I. INTRODUCTION

Power dissipation is the major constraint when it comes to portability feature of a device. Because any consumer demands for more features and extended battery life and also customers want smaller and sleeker devices. This requires high levels of Silicon integration in advanced processes, but advanced processes have inherently higher leakage current. So there is a need to bother more on reducing leakage current to reduce power consumption. An integrated low power methodology requires optimization at all design abstraction layers as mentioned below[1].

1.System:Partitioning,Powerdown
2.Algorithm:Complexity,Concurrency,Regularity
3.Architecture:Parallelism,Pipelining,Redundancy,Data Encoding
4.CircuitLogic:LogicStyles,EnergyRecovery,Transistor Sizing
5. Technology: Threshold Reduction, Multithreshold Devices.

In DSP, DFT plays a major role. FFT is a method for computing the discrete fourier transform (DFT) with reduced number of calculations[2]. The computational efficiency is achieved if we adopt a divide and conquer approach. But still applying various techniques like pipelining, parallelism,efficient arithemetic algorithms can reduce the complexity of FFT further. As the trade off term comes in to picture all the time during optimization, according to the requirement various techniques can be employed to meet the desired characteristics like low power consumption, high speed etc..,

In this paper to achieve low power consumption at the architecture level, pipelined architecture is employed which increases throughput as well as reduces the power consumption with one butterfly unit at each stage. As it is well known that multiplier is the one which consumes much of the power, Gauss complex multiplication method is used in calculating product of complex numbers which uses less number of multiplications compared to the normal method. Meanwhile Vedic multiplier is used as it gives high speed and low power consuming multiplier. So,starting from the architecture level of the FFT till the single multiplier internal calculations are taken care so that at each possible level various optimal techniques are employed for the designing of optimal FFT processor.

II. VEDIC MATHEMATICS

India has created numerous mathematicians whose disclosures have profited the world[3]. One such mathematician was Swami Bharati Krishna Tirtha (1884 – 1960). As an understudy of the Vedas from 1911 – 1918, he discovered few Sutras (axioms or word-formulae)[4]. After broad research on these sutras, he recreated 16 Sutras and 13 Upa-sutras utilizing which numerical figurings is possible in less difficult and faster ways[1]. In Vedic Mathematics, dissimilar to the ordinary strategies, there are numerous approaches to touch base at an answer for an issue[5]. This provides for us the freedom to pick the procedure most advantageous for us. This is the magnificence of Vedic Mathematics; it can be seen effortlessly by individuals of any gauge[3]. It improves the capacity to approach and take care of any numerical.The following are the sixteen sutras:

1. Ekadhikina Purvena which mean “next more than the previous one by one”.

2. Nikhilam Navatashcaramam Dashatah which mean “All from 9 and the last from 10”.

3. Urdhva-Tiryagbyham which mean “Vertically and crosswise”.

4. Paraavartya Yojayet which mean “ Transpose and adjust”.

5. Shunyam Saamyasamuccaye which mean “The sum is zero when the sum is same”.

6 . (Anurupye) Shunyamanyat which mean “If one is in ratio, the other is zero”.

7. Sankalana-vyavakalanabhyam which mean “By addition and by subtraction”.

8 . Puranapuranabyham which mean “By the completion or non-completion”.

9 . Chalana-Kalanabyham which mean “Differences and Similarities”.

10. Yaavadunam which mean “Whatever the extent of its deficiency”.

11. Vyashtisamanstih which mean “part and Whole”.

12. Shesanyankena Charamena which mean “The remainders by the last digit”.
13. Sopaantyadvayamantyam which mean “ The ultimate and twice the penultimate”.
14. Ekanyunena Purvena which mean “By one less than the previous one”.

15. Gunitasamuchyah which mean “The product of the sum is equal to the sum of the product”.

16. Gunakasamuchyah which mean “The factors of the sum is equal to the sum of the factors”.

Now coming to the urdhva thiryagbhyam sutra the multiplication calculation is carried out as below: Following example represents the 3x3 multiplier using vertically and crosswise method.

```
   2 3 3
  x2 1 6
---------
 4 8 1 1 8
   2 2 1
---------
 5 0 3 2 8
---------
```

Steps:
i) 3 x 6 = 18 : 1, the left most bit is placed below the second digit.
ii) (3 x 6) + (3 x 1) = 18 + 3 = 21 ; 2, the left most bit is placed below third digit.
iii) (2 X 6) + (3 X 1) + (3 X 2) = 12 + 3 + 6 = 21 ; 2, the left most digit is placed below fourth digit.
iv) (2 X 1) + (3 X 2) = 2 + 6 = 8; 0, the leftmost bit is placed below fifth digit.
v) (2 X 2) = 4.
vi) Respective digits are now added.

So as explained above, vedic multiplier computes the output bits very fastly.It just involves two steps. In the first step expression of each bit position of total output size will be directly calculated from the presolved expression for that particular bit position[5]. In the second step the carries which ever resulted due to expressions will be added to the next bit position which gives the final output of the multiplier. Due to its very simple computation we can achieve high speed as well as low power featured multiplier.

III. SINGLE PRECISION FLOATING POINT NUMBERS

The IEEE 754 standard represents a binary32 format as follows:

Signbit : 1bit - Sign bit determines the sign of the number, which is the sign of the significand as well.

Exponent width: 8 bits - Exponent is either an 8 bit signed integer from −128 to 127 (2's Complement) or an 8 bit unsigned integer from 0 to 255 which is the accepted biased form in IEEE 754 binary32 definition. If the unsigned integer format is used, the exponent value used in the arithmetic is the exponent shifted by a bias – for the IEEE 754 binary32 case, an exponent value of 127 represents the actual zero (i.e. for 2^{e-127} to be one, e must be 127).

Significand precision: 24 bits (23 explicitly stored). The true significand includes 23 fraction bits to the right of the binary point and an implicit leading bit (to the left of the binary point) with value 1 unless the exponent is stored with all zeros. Thus only 23 fraction bits of the significand appear in the memory format but the total precision is 24 bits). This gives from 6 to 9 significant decimal digits precision.The bits representation in single precision are laid out as shown in Fig-1:

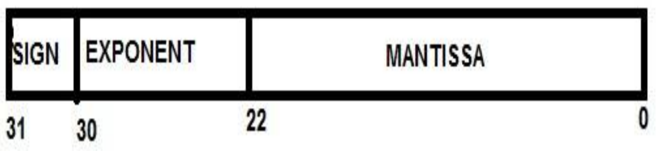


Figure 1: Single precision floating point representation

This single precision representation of digits gives accurate outputs which is very essential in digital signal processing. So the multiplications and additions involved in algorithm are computed using this floating point representation for high accuracy.

IV. GAUSS COMPLEX MULTIPLICATION TECHNIQUE

Complexmultiplication generally involves four multiplications and two additions as shown below:.

$$(a+bi)(c+di)=(ac-bd)+(bc+ad)i.$$

The product of $(a + bi)$ and $(c + di)$ can be calculated in the following way.

$k_1 = c * (a + b)$
$k_2 = a * (d - c)$
$k_3 = b * (c + d)$
Real part $= k_1 - k_3$
Imaginary part $= k_1 + k_2$.

This algorithm uses only three multiplications, rather than four, and five additions or subtractions rather than two[2]. As multiplier consumes more power in a circuitry, the maximum possible reduction of multipliers usage results in an efficient low power design. As FFT algorithm comprises many number of complex multiplications in the computation of final output reducing one multiplication at each single complex multiplication going to show a very significant change in the performance of the design.

V. RADIX-2 FFT

The Fast Fourier Transform is a method for computing the discrete Fourier transform (DFT) with reduced number of calculations[6]. The computational efficiency is achieved if we adopt a divide and conquer approach[6].

For Radix-2 FFT, the value of N should be such that, N = 2 power m, so that the N- point sequence is decimated in to 2-point DFT for each decimated sequence is computed. From the results of 2-point DFTs, the 4-point DFTs can be computed. From the results of 4-point DFTs, the 8-point DFTs can be computed and so on, until we get N- point DFT.

In radix -2 FFT, N=2 power m, and so there will be m stages of computations, where the total number of complex additions are reduced to NlogN and total number of complex multiplications are reduced to (N/2)logN[5]. The following Fig-2 shows the signal flow graph of Radix-2 16-point DIF-FFT.

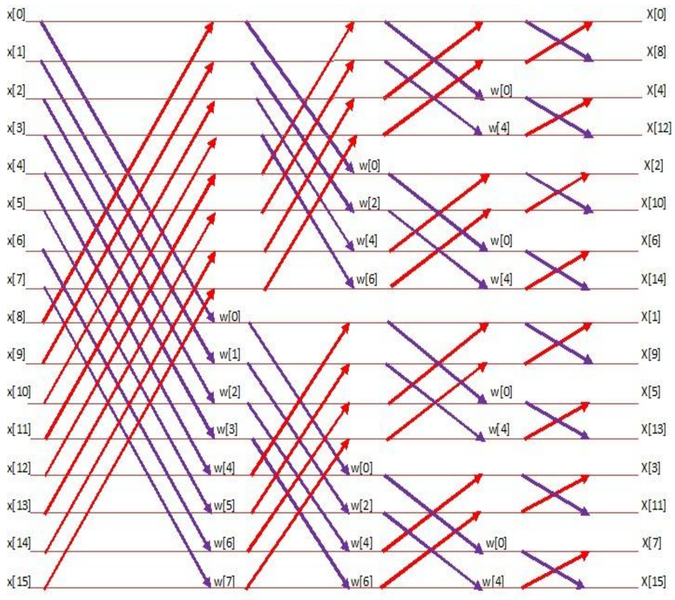


Figure 2. Signal flow graph of 16-point Radix-2 DIF-FFT

VI. RADIX-2 PIPELINED ARCHITECTURE

Pipelining transformation leads to a reduction in the critical path, which can be exploited to either increase the clock speed or sample speed or to reduce power consumption at same speed[7]. Pipelining reduces the effective critical path by introducing pipelining latches along the datapath. The following Fig-3 shows the pipelined architecture of FFT processor

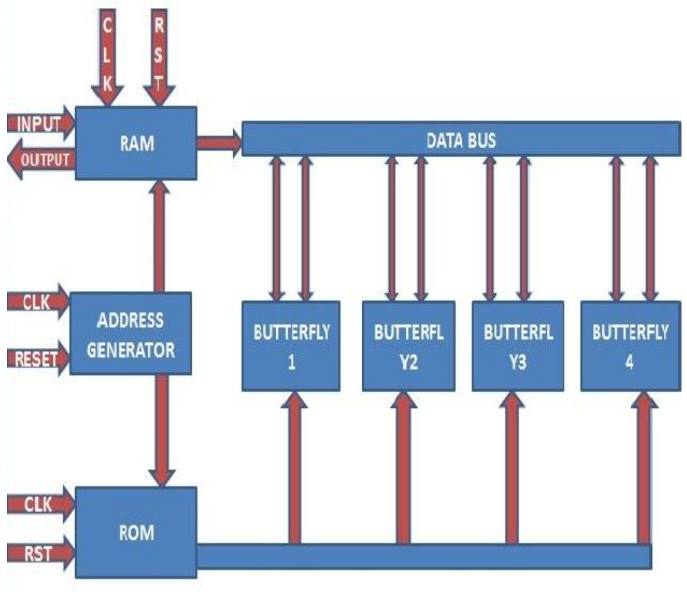


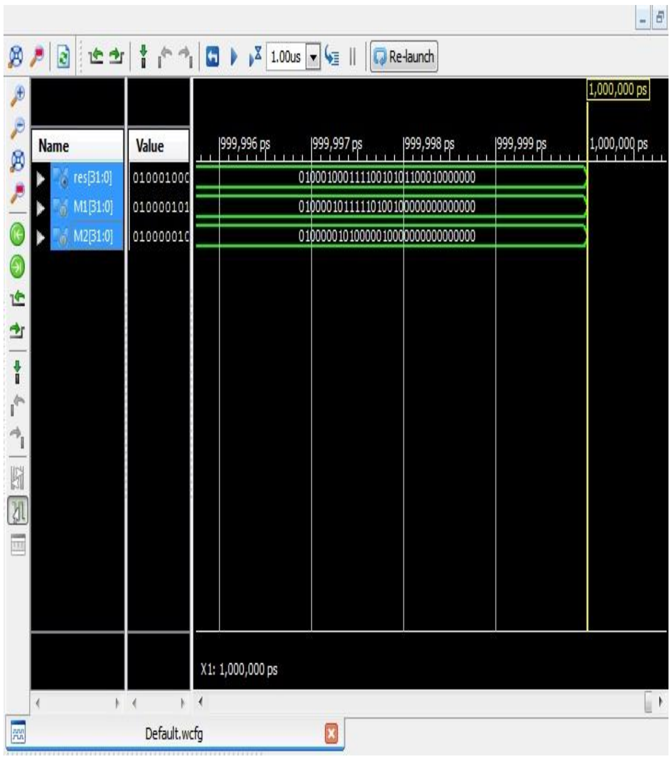
Figure 3. Pipelined architecture of 16-point FFT

In this pipelined architecture area reduction technique "IN-place addressing" method is used i.e, as outputs and inputs are required to be stored in RAM only single RAM is used for storing both the inputs and outputs. When the inputs are fetched from the RAM, that particular registers will be empty.So that we can utilize that registers for storing the outputs generated during that period.

So coming to the description part of the pipelined architecture, it contains RAM which stores inputs and outputs according to IN-place addressing method.ROM stores the Twiddle factors which are generally the constants.Next one of the main block which plays crucial role in architecture is the address generator which is responsible for the synchronised data flow and twiddle factors to the butterfly units.Butterfly units are the main units which actually processes the input data.

VII. RESULTS

This section shows all the simulated and synthesized results. These simulations are done using Xilinx ISE tool and synthesis reports of power, area and timing are extracted from Cadence RC tool. In this comparision tables category: traditional refers to the normal conventional FFT i.e,parallel FFT, pipelined refers to only pipelined architecture of FFT and optimized FFT refers to the pipelined FFT with Gauss complex multiplication method and vedic multiplier.


```
Name        Value       |999,996 ps   |999,997 ps   |999,998 ps   |999,999 ps   1,000,000 ps
res[31:0]   010001000   01000100011110010101100010000000
M1[31:0]    010000101   01000010111110100100000000000000
M2[31:0]    010000010   01000001010000010000000000000000

X1: 1,000,000 ps
Default.wcfg
```

Figure 4. Simulated output of floating point vedic multiplier

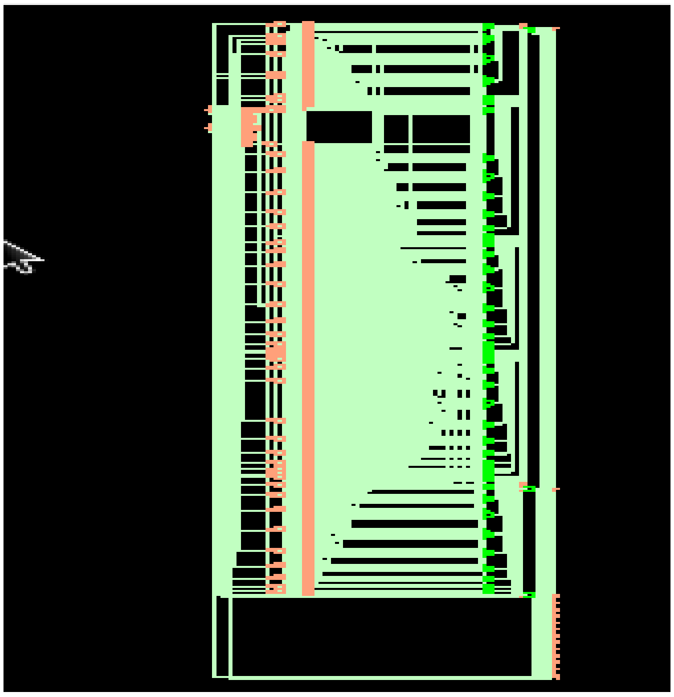
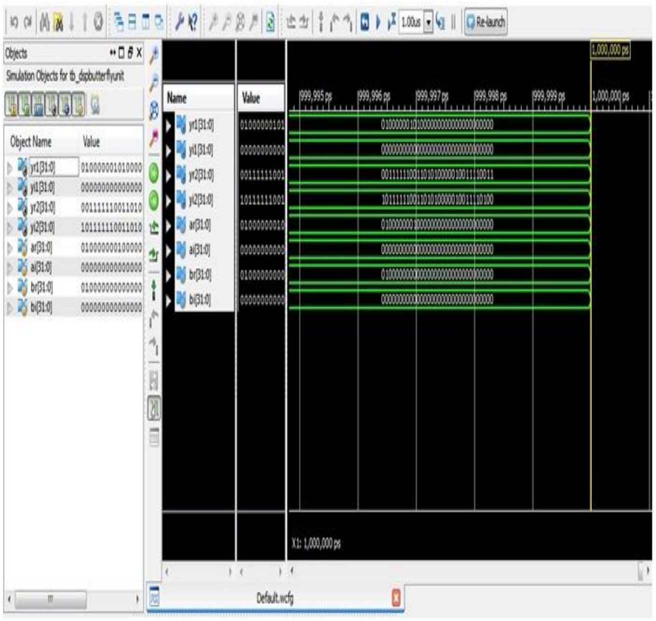


Figure 5. Synthesis outcome of floating point multiplication along with Vedic multiplier


```
Objects
Simulation Objects for tb_dspbutterflyunit

Object Name   Value
yr1[31:0]     010000001010000
yi1[31:0]     000000000000000
yr2[31:0]     001111110011010
yi2[31:0]     101111110011010
ar[31:0]      010000000100000
ai[31:0]      000000000000000
br[31:0]      010000000000000
bi[31:0]      000000000000000

Name        Value         |999,995 ps  |999,996 ps  |999,997 ps  |999,998 ps  |999,999 ps  1,000,000 ps
yr1[31:0]   01000000101   01000000101000000000000000000000
yi1[31:0]   00000000000   00000000000000000000000000000000
yr2[31:0]   00111111001   00111111001101010000010011110011
yi2[31:0]   10111111001   10111111001101010000010011110100
ar[31:0]    01000000010   01000000010000000000000000000000
ai[31:0]    00000000000   00000000000000000000000000000000
br[31:0]    01000000000   01000000000000000000000000000000
bi[31:0]    00000000000   00000000000000000000000000000000

X1: 1,000,000 ps
Default.wcfg
```

Figure 6. Simulated output of butterfly unit with Gauss multiplication algo

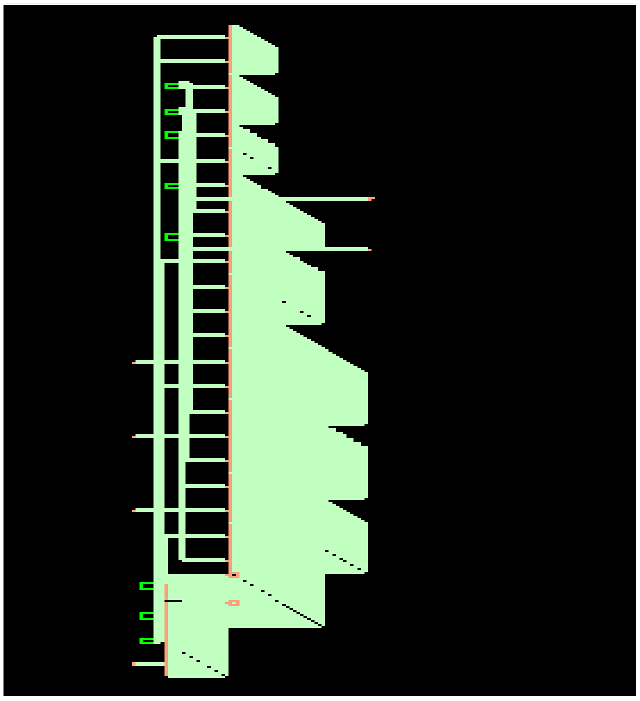


Figure 7. Synthesis outcome of Gauss complex multiplication method

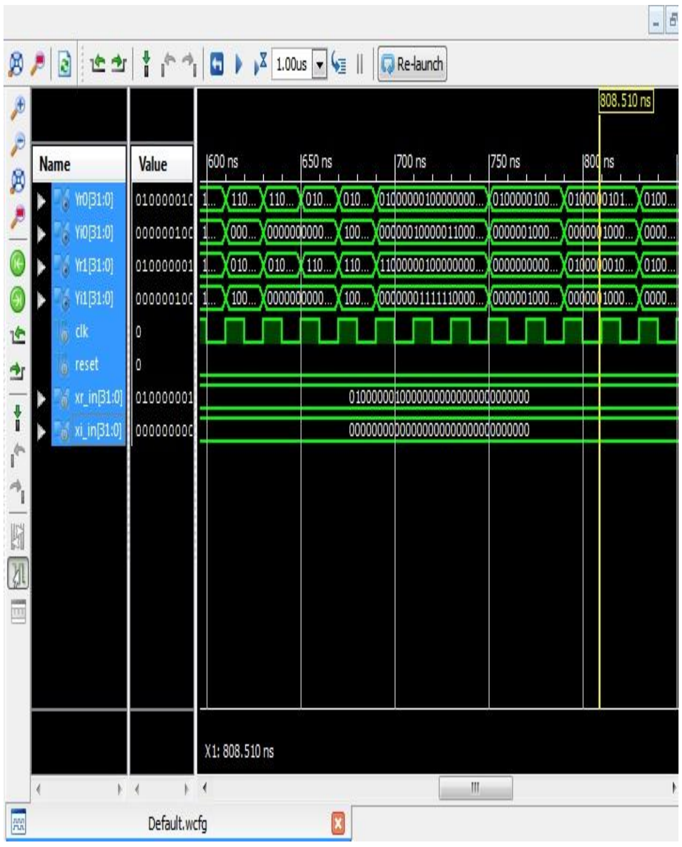

```
Name          Value
Yr0[31:0]     010000010
Yi0[31:0]     000000100
Yr1[31:0]     010000001
Yi1[31:0]     000000100
clk           0
reset         0
xr_in[31:0]   010000001
xi_in[31:0]   000000000
```
```
xr_in[31:0]  01000000100000000000000000000000
xi_in[31:0]  00000000000000000000000000000000
```
X1: 808.510 ns

Figure 8. Simulated output of optimized pipelined fft processor

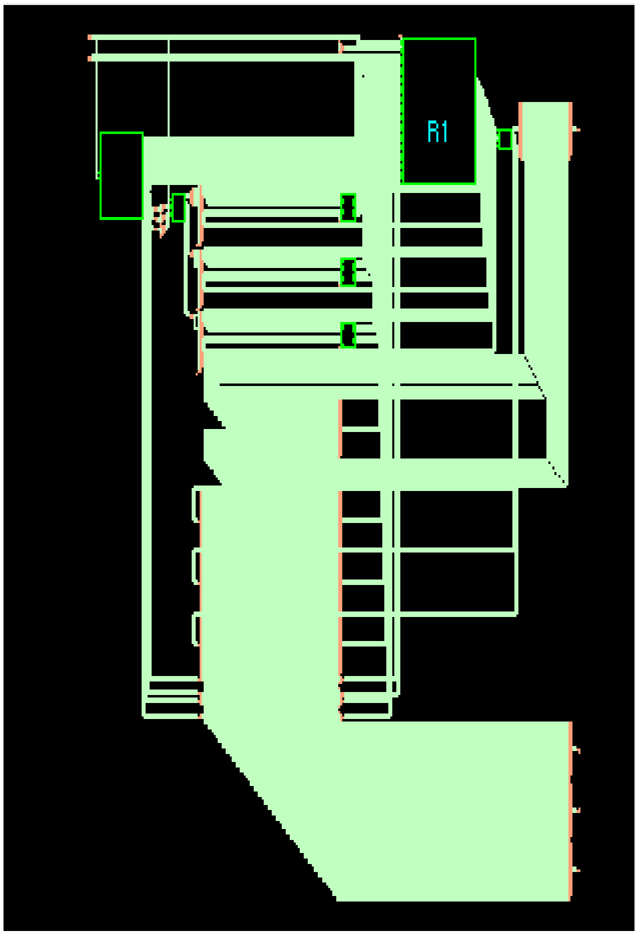
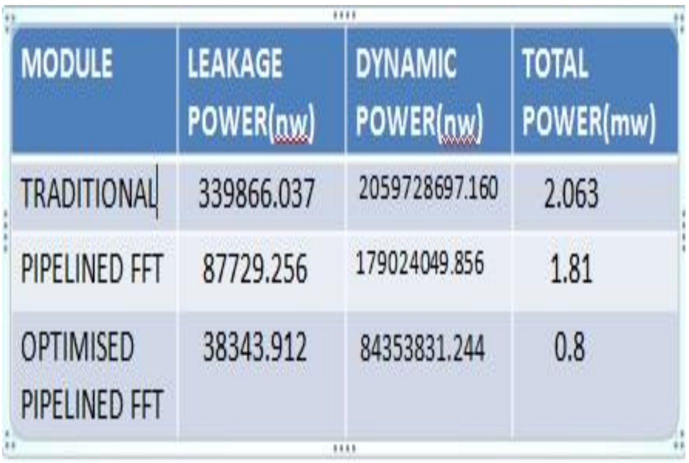


Figure 9. synthesis outcome of pipelined FFT processor

TABLE I
Comparison of performance of FFTs based on power consumption



MODULE	LEAKAGE POWER(nw)	DYNAMIC POWER(nw)	TOTAL POWER(mw)
TRADITIONAL	339866.037	2059728697.160	2.063
PIPELINED FFT	87729.256	179024049.856	1.81
OPTIMISED PIPELINED FFT	38343.912	84353831.244	0.8

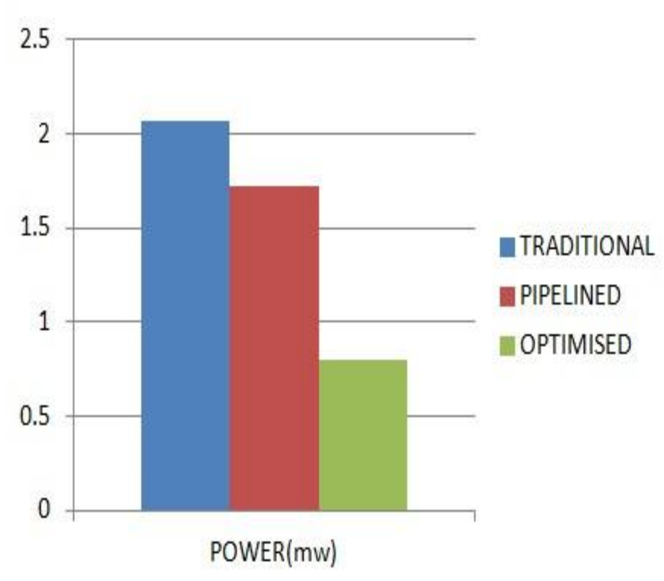


Figure 10. Bar chart representing power variation for different proposed FFT implementations

TABLE II
Comparison of performance of FFTs based on area occupancy

MODULE	CELLS	CELL AREA
TRADITIONAL	595090	10724766
PIPELINED FFT	160078	2669862
OPTIMISED PIPELINED FFT	72729	1350063

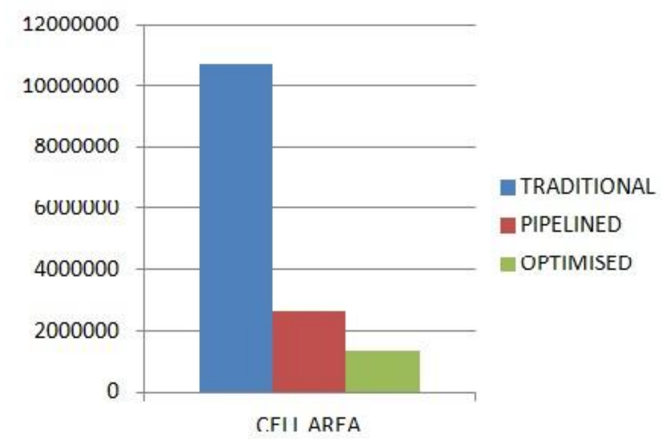


Figure 11. Bar chart representing area of different proposed FFT implementations

TABLE III
Comparison of performance of FFTs based on operating frequency

MODULE	CLOCK PERIOD	OPERATING FREQUENCY
TRADITIONAL	200ns	5MHz
PIPELINED FFT	55ns	18MHz
OPTIMISED PIPELINED FFT	25ns	40MHz

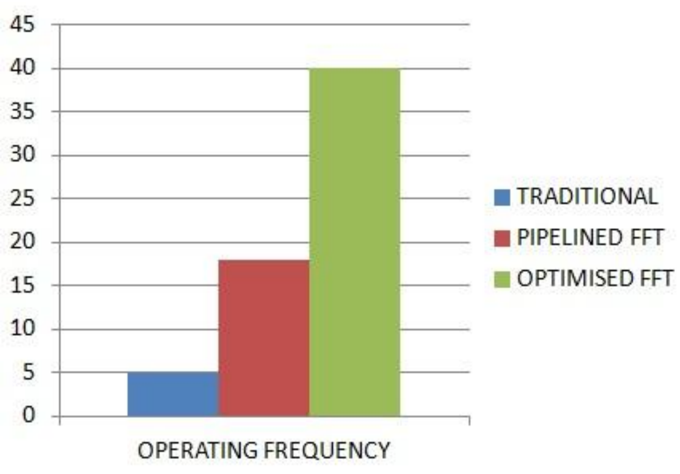


Figure 12. Bar chart representing variation in operating frequency of proposed FFTs.

VII. CONCLUSION

From the results above, it is clear that pipelined architecture is giving better performance than the traditional FFT in terms of area, power and operating frequency. Further when low power technique like Gauss complex multiplication method is used in pipelined architecture, the power consumption of FFT decereased significantly and when vedic multiplier (Urdhava thiryagbhyam) is used the throughput of fft increased to a large extent due to its fast computation method of multiplication. So employing these pipelining ,Gauss complex method and vedic multiplier the fft performance increased in operating frequency as well as consumes less power with less hardware requirement. As these proposed techniques (Gauss and vedic multiplication methods) reduce the complexity due to multiplications, they can be applied in any FFT Radix-r algorithm for efficient performance as each algo obviously consists of multiplications.

REFERENCES

(1) G.Yeap, Practical Low Power Digital VLSI Design, Springer Science+Business media, LLC.

(2) J.G. Proakis and D.G. Monalkies,1988, "Digital Signal Processing. " Macmillian.

(3) High speed and area efficient vedic multiplier by kunchigi.v,Jawaharlal Nehru Technol. Univ., Hyderabad, India kulakarni.l,. ; Kulkarn.s.

(4) Saha.P,Banerjee.A,Bhattacharyya.P,Dandapat.A,"High speed ASIC design of complex multiplier using VedicMathematics,"Students' Technology Symposium (TechSym), 2011 IEEE , vol., no., pp.237-241, 14-16 Jan. 2011.

(5) HumanTharafu M.C. Jayalaxmi. H. Renuka R.K.,Ravishankar. M.,2007, "A. high speed block convolution using Ancient Indian Vedic Mathematics," IEEE International conference on computational intelligence and multimedia applications.

(6) Ahmed Saeed, M.Elbably, G. Abdelfadeel and M.I. Eladawy "Efficient FPGA implementation of FFT/IFFT Processor". International Journal of Circuits, Systems and Signal Processing, Issue 3, Volume 3,2009 S Salivahanan, A.Vallavaraj and C.Gnanapriya, Digital Signal Processing.

(7) Samir Palnitkar, Verilog HDL- A Guide to Digital Design and Synthesis,IEEE 1364-2001 Compliant.