International Journal of Scientific Engineering and Research (IJSER)
ISSN (Online): 2347-3878
Index Copernicus Value (2015): 62.86 | Impact Factor (2015): 3.791

Implementation of High Speed FFT using Reliable Multiplier with AHL Circuit

S. Arthi[1], G. Prathipa[2]

[1]Student (M.E- VLSI), Dept. of ECE, Karpaga Vinayaga College of engineering and technology, Anna University, Chennai, India
[2]Assistant Professor, Dept. of ECE, Karpaga Vinayaga College of engineering and technology, Anna University, Chennai, India

Abstract: ***Fast Fourier Transform (FFT) is used to convert a signal from Time domain to frequency domain. FFT and IFFT algorithm plays an important role in design of digital signal processing. An FFT is an algorithm that speeds up the calculation of a DFT.FFT is a DFT for speed. The entire purpose of an FFT is to speed up the calculations. This paper describes the design of Decimation in Time-Fast Fourier Transform (DIT-FFT). The proposed design is implemented with radix-2, based 4 point FFT. Whereas digital multipliers are among the most critical arithmetic functional units. The overall performance of these systems depends on the throughput of the multiplier. Here a reliable multiplier with adaptive hold logic is used. This approach reduces the multiplicative complexity which exists in conventional FFT implementation. For the number representation of FFT fixed point arithmetic has been used. The overall performance of the FFT is based on the throughput of the Multiplier. Here the multiplier with AHL is used to reduce the power consumption and to increase the speed of the FFT. The design is implemented using Verilog HDL language.***

Keywords: DIT-FFT, Complex multiplication, Verilog, Radix-2, Adaptive Hold Logic

1. Introduction

FFT and IFFT commonly used algorithm for processing signals. It can be used for WLAN, image process, spectrum measurements, radar and multimedia communication services. Now a days, FFT processors were using in wireless communication systems that are having fast execution and low power consumption. These are some most important constraints of FFT processor. Complex multiplication is main arithmetic operation used in FFT/IFFT blocks. This is the main issue in processor. It is time consuming and it consumes a large chip area and power. When large point FFT is to be designed, it increases the complexity. To reduce the complexity of the multiplication, there are two methods one simple method is to real and constant multiplications take the place of complex multiplication. The other method is non-trivial complex multiplication is wipe out by the twiddle factors and fulfils the processing with no complex multiplication. In our work FFT algorithm is implemented in radix 2. The basic idea of these algorithms is to divide the N-point FFT into smaller ones until two point FFT is obtained. Hence the algorithm is called radix-2 algorithm. Here the multiplier used is the reliable multiplier design using adaptive hold logic. In this again we use Row/Column Bypassing Techniques to reduce the Dynamic power and delay for the multiplication process. Vedic Mathematics is the ancient system of mathematics which has a unique technique of calculations based on 16 Sutras. Employing these techniques in the computation algorithms of the coprocessor will reduce the complexity, execution time, area, power etc. In this paper reconfigurable FFT is proposed to design by Vedic mathematics.
Digital multipliers are the core components of all the digital signal processors (DSPs) and the speed of the DSP is largely determined by the speed of its multipliers. They are indispensable in the implementation of computation systems realizing many important functions such as fast Fourier transforms (FFTs) and multiply accumulate (MAC). Two most common multiplication algorithms followed in the digital hardware are array multiplication algorithm and Booth multiplication algorithm. The computation time taken by the array multiplier is comparatively less because the partial products are calculated independently in parallel. The delay associated with the array multiplier is the time taken by the signals to propagate through the gates that form the multiplication array. Booth multiplication is another important multiplication algorithm. Large booth arrays are required for high speed multiplication and exponential operations which in turn require large partial sum and partial carry registers. Multiplication of two n-bit operands.

Discrete Fourier Transform (DFT)

The Fourier transform is mathematical method of changing time representation of signal into frequency representation. It transforms one function from time domain to frequency domain. The DFT of a input sequence x[n] can be computed using the formula given:

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

Where $W_N = e^{-j2\pi/N}$

Radix 2 DIT – FFT Algorithm

The basic module for implementation is butterfly module which is shown in the Figure 1.

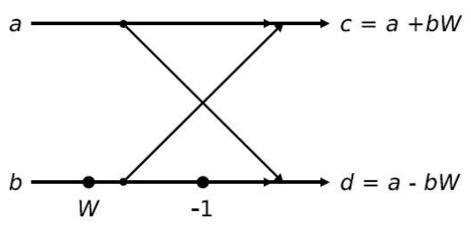


Figure 1: Radix-2 Structure

There are two inputs called a, b and two outputs c, d and twiddle factor W. Output is as follows:

c= a + bW (3)
d = a – bW (4)

With these butterfly units we can built whole FFT structure. If N is the input for FFT then stages are required, each stage requires N/2 butterflies. As we can see from above Figure that one butterfly unit requires 1 complex multiplier and 2 adders for executing single butterfly. For every DIT-FFT radix – 2 algorithms with N input sequence requires N/2 multipliers and N adders.

Number Representation

For number representation of both real as well as imaginary fixed point scheme is followed so that we can reduce the complexity of using floating point arithmetic. The twiddle factor used is in complex form real and imaginary. To represent this number we are multiplying these numbers by scaling factor which is where s N. So that twiddle factor is rounded up in integer number. For complex multiplication we require twiddle factor magnitude and sign bit so s+1 bit are required to represent twiddle factor. As the input given to the design can also be in floating form then we can apply the same scheme of rounding up input in the integer. As we are scaling up the input or twiddle factor we have to scale down the signals at the output and we have to accept some rounding errors. Simple way for scaling down is by multiplying or using shifting operation. It is as simple as we are multiplying one no with something and dividing the same number we will get the original number.

Complex Multiplier

Here FFT is implemented by using the proposed multiplier. An Aging-aware reliable multiplier design with adaptive hold logic (AHL) is used for multiplication. This multiplier is based on the variable-latency technique and can adjust the AHL circuit to achieve reliable operation under the influence of NBTI and PBTI effects. To be specific, the contributions of this multiplier are summarized as follows:

1) Variable-latency multiplier architecture with an ahlcircuit.
2) A reliable multiplier design method that is suitable for large multipliers. The experiment is performed in 4, 16, 32 and 64-bit multiplications.
3) An FFT was designed by using this multiplier.

Negative bias temperature instability

The negative bias temperature instability (NBTI) occurs when a pMOS transistor is under negative bias (Vgs =− Vdd). In this situation, the interaction between inversion layer holes and hydrogen-passivated Si atoms break the Si–H bond generated during the oxidation process, generating H or H2 molecules. When these molecules diffuse away, interface traps are left. The accumulated interface traps between silicon and the gate oxide interface result in increased threshold voltage (Vth), reducing the circuit switching speed. When the biased voltage is removed, the reverse reaction occurs, reducing the NBTI effect.

Positive bias temperature instability

The PBTI effect is much smaller on oxide/polygate transistors, and therefore is usually ignored. However, for high-k/metal-gate nMOS transistors with significant charge trapping, the PBTI effect can no longer be ignored. In fact, it has been shown that the PBTI effect is more significant than the NBTI effect on 32-nm high-k/metal-gate processes.

2. Existing System

1) Array Multiplier

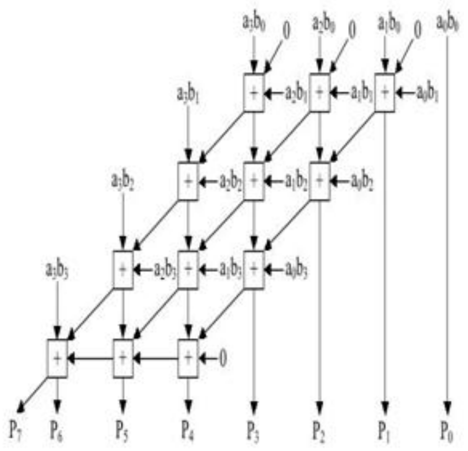


Figure 2: 4x4 normal array multiplier

The AM is a fast parallel AM and is shown in Figure 2. The multiplier array consists of (n−1) rows of carry save adder (CSA), in which each row contains (n−1) full adder (FA) cells. Each FA in the CSA array has two outputs: 1) the sum bit goes down and 2) the carry bit goes to the lower left FA. The last row is a ripple adder for carry propagation. The FAs in the AM are always active regardless of input state. Multiplier circuit based on add and shift algorithm. Each partial product is generated by the multiplication of the multiplicand with one multiplier bit. The partial product are shifted according to their bit orders and then added. The addition can be performed with normal carry propagate adder. N-1 adders are required where N is the multiplier length. A 4bit * 4bits booth encoded Wallace tree multiplier are implemented in verilog to demonstrate the proposed multiplier.

2) Column-Bypassing Multiplier

A column-bypassing multiplier is an improvement on the normal array multiplier (AM). The AM is a fast parallel AM and is shown in Figure 3. The multiplier array consists of (n − 1) rows of carry save adder (CSA), in which each row contains (n − 1) full adder (FA) cells. Each FA in the CSA array has two outputs: 1) The sum bit goes down and 2) The carry bit goes to the lower left FA. The last row is a ripple adder for carry propagation. The

FAs in the AM are always active regardless of input states. A low-power column-bypassing multiplier design is proposed in which the FA operations are disabled if the corresponding bit in the multiplicand is 0.Figure 3. shows a 4×4 column-bypassing multiplier.

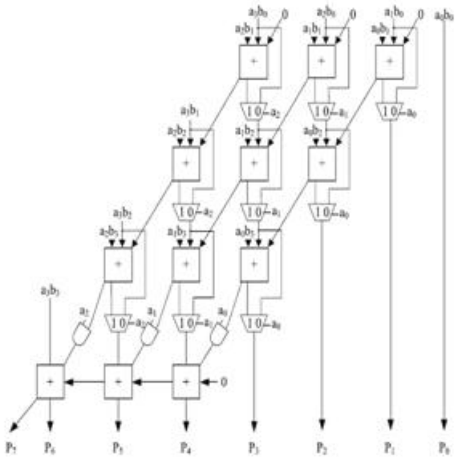


Figure 3: 4x4 column-bypassing multiplier

The multiplicand bit ai can be used as the selector of the multiplexer to decide the output of the FA, and ai can also be used as the selector of the tri-state gate to turn off the input path of the FA. If ai is 0, the inputs of FA are disabled, and the sum bit of the current FA is equal to the sum bit from its upper FA, thus reducing the power consumption of the multiplier. If ai is 1, the normal sum result is selected. Column Bypassing with reference to multiplier means turning off some columns in the multiplier array whenever certain multiplicand bits are zero. In this technique, during working, the operations in a column can be disabled if the corresponding bit in the multiplicand is 0, to save the power. This technique is totally depended on the number of zeroes in the multiplicand bits.

3) Row-Bypassing Multiplier

A low-power row-bypassing multiplier is also designed to reduce the activity power of the AM. The operation of the low-power row-bypassing multiplier is similar to that of the low power column-bypassing multiplier, but the selector of the multiplexers and the tristate gates use the multiplicator. Figure 4 is a 4×4 row-bypassing multiplier. Each input is connected to an FA through a tri-state gate.

When the inputs are 11112 * 10012, the two inputs in the first and second rows are 0 for FAs. Because b1 is 0, the multiplexers in the first row select ai b0 as the sum bit and select 0 as the carry bit.

The inputs are bypassed to FAs in the second rows, and the tri-state gates turn off the input paths to the FAs. Therefore, no switching activities occur in the first-row FAs; in return, power consumption is reduced. Similarly, because b2 is0, no switching activities will occur in the second-row FAs. However, the FAs must be active in the third row because the b3 is not zero.

There are two advantages of this approach. First, it eliminates the extra correcting circuit. Second, the modified FA is simpler than that used in the row-by passing multiplier**.**

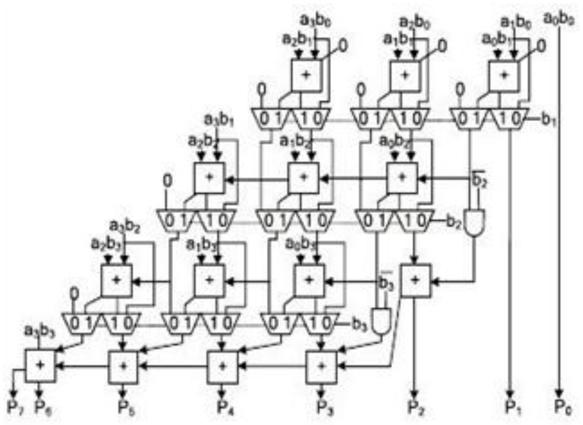


Figure 4: 4x4 row bypassing multiplier

3. Proposed System

a) Reliable Multiplier Design with Adaptive Hold Logic

Digital multipliers are among the most critical arithmetic functional units. The overall performance of these systems depends on the throughput of the multiplier. Meanwhile, the negative bias temperature instability effect occurs when a pMOS transistor is under negative bias (V_{gs} = -V_{dd}), increasing the threshold voltage of the pMOS transistor, and reducing multiplier speed. A similar phenomenon, positive bias temperature instability, occurs when an nMOS transistor is under positive bias. Both effects degrade transistor speed, and in the long term, the system may fail due to timing violations. Therefore, it is important to design reliable high-performance multipliers. In this paper, we propose an aging-aware multiplier design with a novel adaptive hold logic (AHL) circuit. The multiplier is able to provide higher throughput through the variable latency and can adjust the AHL circuit to mitigate performance degradation that is due to the aging effect. Moreover, the proposed architecture can be applied to a column or row-bypassing multiplier. The experimental results show that our proposed architecture with 16 × 16 and 32 × 32 column-bypassing multipliers can attain up to 62.88% and 76.28% performance improvement, respectively, compared with 16×16 and 32×32 fixed-latency column-bypassing multipliers. Furthermore, our proposed architecture with 16 × 16 and 32 × 32 row-bypassing multipliers can achieve up to 80.17% and 69.40% performance improvement as compared with 16×16 and 32 × 32 fixed-latency row-bypassing multipliers.

b) Razor flipflop:

Razor flip-flops can be used to detect whether timing violations occur before the next input pattern arrives. A 1-bit Razor flip-flop contains a main flip-flop, shadow latch, XOR gate, and multiplexer. The main flip-flop catches the execution result for the combination circuit using a normal clock signal, and the shadow latch catches the execution result using a delayed clock signal the Razor flip-flop will set the error signal to 1 to notify the system to re execute the operation and notify the AHL circuit that an error has occurred.

c) Vedic Multiplier Design With Adaptive Hold Logic

Vedic multiplier using AHL. It has the total architecture and describes the operations of each block. The architecture which includes 4x4 Vedic multiplier, razor flip-flop and an AHL circuit. The design of Vedic multiplier using AHL circuit. The AHL circuit can determine which input pattern to need one or two cycles and then the circuit decides to make suitable the judging block to reduce the errors in the circuits.

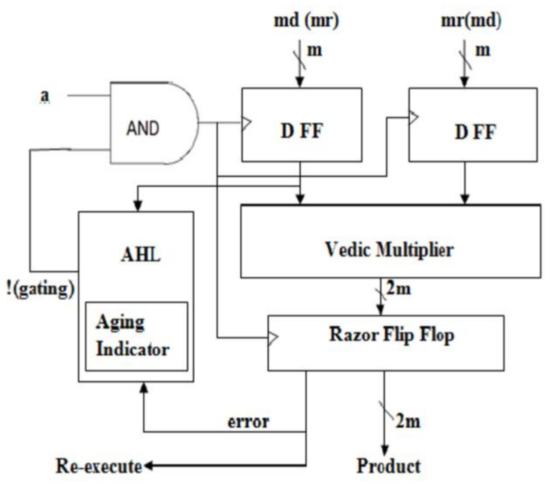


Figure 5: Proposed Multiplier Architecture

Adaptive Hold Logic:

An Adaptive Hold Logic (AHL) circuit is proposed which will reduce the aging effects. The Adaptive Hold Logic (AHL) circuit can decide whether the input patterns require one or two cycles and can adjust the judging criteria to ensure that there is minimum performance degradation after considerable aging occurs. The detailed architecture of the Adaptive Hold Logic (AHL) model is shown in Figure no.1

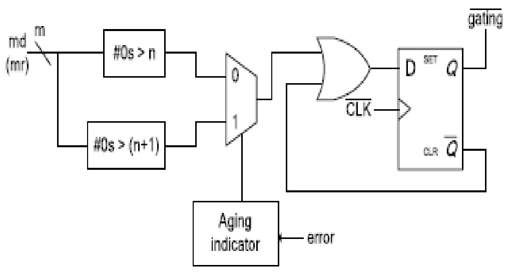


Figure 6: Diagram of Adaptive hold logic

The AHL circuit adjusts itself to mitigate performance degradation which is caused due to aging effects. This proposed work can be used in various VLSI applications to perform operations like addition, multiplication etc. The AHL circuit will produce output which will be both power and area efficient. The results will be in terms of Latency and Power consumption which will improve the circuit performance.

Aging indicator

The Aging Indicator indicates that whether the circuit has suffered significant performance degradation due to aging effects. The aging effect is not significant in the beginning, so the aging indicator produces output as 0. The Aging Indicator is implemented in a counter that counts the number of errors over a certain amount of operations. These operations may be multiplication or addition. It resets to zero at the end of those operations.

Architecture of 4x4 bit Vedic multiplier:

The most important arithmetic operation in signal processing applications and inside the Processor. As speed is always a major requirement in the multiplication operation, increase in speed can be achieved by reducing the number of steps in the computation process. The speed of multiplier determines the efficiency of such a system. In any system design, the three main constraints which determine the performance of the system are speed, area and power requirement. Vedic mathematics was reconstructed from the ancient Indian scriptures (Vedas) by Swami Bharati Krishna Tirthaji Maharaja (1884-1960) after his eight years of research on Vedas. Vedic mathematics is mainly based on sixteen principles or word-formulae which are termed as sutras. This is a very interesting field and presents some effective algorithms which can be applied to various branches of engineering such as computing and digital signal processing. Integrating multiplication with Vedic Mathematics techniques would result in the saving of computational time

The 4x4 multiplication has been done in a single line in Urdhva Tiryagbhyam sutra, where as in shift and add (conventional) method, four partial products have to be added to get the result. Thus, by using Urdhva Tiryagbhyam Sutra in binary multiplication, the number of steps required calculating the final product will be reduced and hence there is a reduction in computational time and increase in speed of the multiplier.

A 4-bit modified multiplier is designed by using the proposed 4 bit adder. The proposed 4x4 multiplier gives a total delay of 12.825 ns which is less when compared to the total delay of existing multiplier architecture.

The 4x4 bit Vedic multiplier is implemented by using four 2x2 Vedic multiplier. To illustrate, 4x4 Vedic multiplication, it have A=A3A2A1A0, B=B3B2B1B0 and the output is S7S6S5S4S3S2S1S0. Let’s A & B is divided into 2 parts A3A2 & A1A0 for A and B3B2 & B1B0 for B by using the basic of vedic multiplication, taking the 2 bit simultaneously in the circuit by using two bit multiplier block. Vedic multiplier using adaptive hold logic. We analyze the Vedic multiplier using AHL as compared to the Vedic multiplier without using AHL. By using AHL, the circuit minimizes the timing waste in the multiplier. So, it reduces the power consumption and delay in the circuit.

The 4x4 bit Vedic multiplier with AHL can achieve the power and delay is significantly reduced as compared to Vedic multiplier without AHL.

The Vedic multiplier and AHL perform their operation at a same time. The multiplicand (multiplicator) have number of zeros, the adaptive hold circuit decide it takes 1 or 2 cycles. After finishing the operation of Vedic multiplier and it goes to razor flip flop. The razor flip-flop checks whether there timing violation. So, if any timing violation occurs it re-executes the operation by using two cycles pattern and it indicate to the AHL. Our architecture minimizes the timing violation of the circuit.

4. Result

The High speed FFT by Vedic multiplier is implemented into the Vertex 2 pro, Device-XC2VP2, package- FG256, speed:-6. vedic mathematics are going to reduced the number of adder and multiplier as compare to the conventional method. The implementation of DIT-FFT using traditional multipliers and the aging aware multiplier are also done using Verilog HDL in Xilinx14.1, and the simulations are observed with ISE simulator. Xilinx Synthesizer is used to analyse the delay.

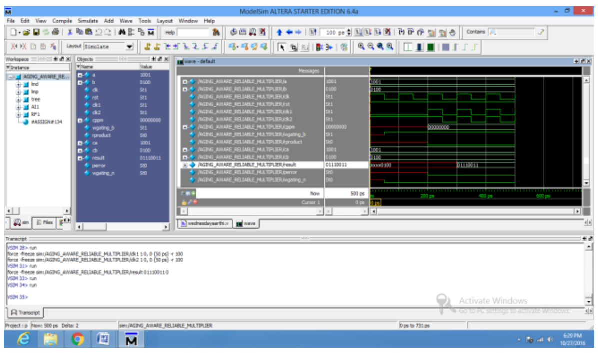


Figure 7: Simulation Result

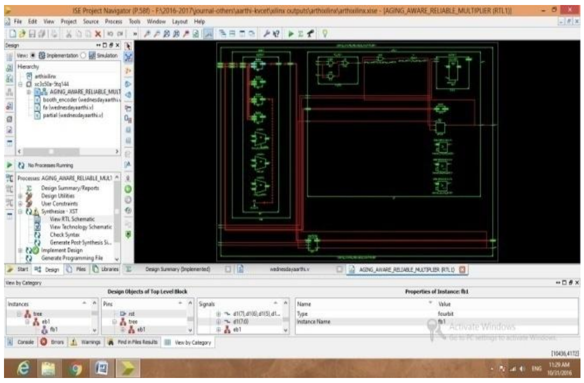


Figure 8: RTL Schematic

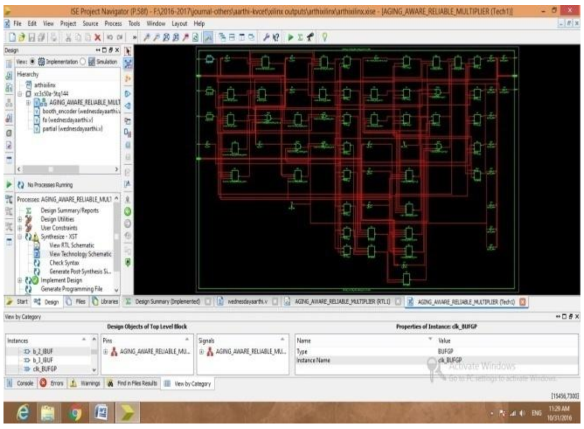


Figure 9: Technology Schematic

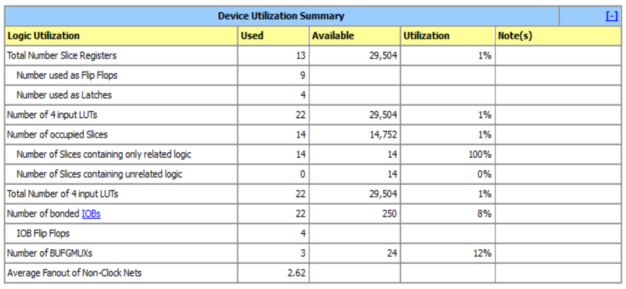
Figure 8 and Figure 9 shows the RTL schematic view and Technology schematic view of the model.

5. Conclusion

In this paper the implementation of high speed FFT is designed to reduce the delay. The FFT which is implemented using the AHL technique where as the AHL Multiplier has three important features. First, its Delay is very less when compared to the other traditional Multipliers. Second, it can provide reliable operations even after the aging effect occurs. The Razor flip-flops detect the timing violations and re-execute the operations using two cycles. Last but not least, the AHL architecture is power efficient and it can also adjust the percentage of one-cycle patterns to minimize performance degradations due to the aging effect. When the circuit is aged, and many errors occur, the AHL circuit uses the second judging block to decide if an input is one cycle or two cycles and hence the timing errors can also be eliminated and can perform the error free operations. The proposed FFT is implemented using AHL architecture in its multiplication process has a great advantage in terms of Delay and hence, Adaptive Hold Logic can be stated as the reliable multiplier technique which can be used in FFTs in harsh environment mostly in aerospace applications etc.

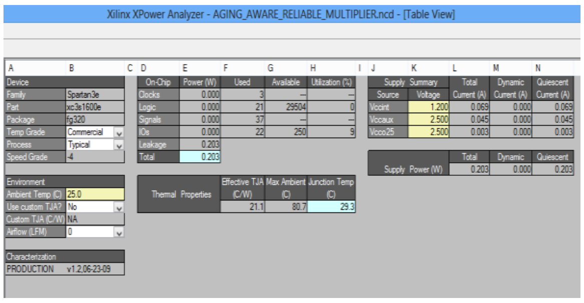
The implementation of radix-2, 4 point DIT-FFT is done using Verilog HDL, as a future scope 8 point, 16 point, etc. DIT-FFTs can be implemented with the help of the aging aware multiplier to get high speed and power efficient devices.

Tabulated parameter area, power and delay:



Device Utilization Summary				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Total Number Slice Registers	13	29,504	1%	
Number used as Flip Flops	9			
Number used as Latches	4			
Number of 4 input LUTs	22	29,504	1%	
Number of occupied Slices	14	14,752	1%	
Number of Slices containing only related logic	14	14	100%	
Number of Slices containing unrelated logic	0	14	0%	
Total Number of 4 input LUTs	22	29,504	1%	
Number of bonded IOBs	22	250	8%	
IOB Flip Flops	4			
Number of BUFGMUXs	3	24	12%	
Average Fanout of Non-Clock Nets	2.62			

Tabulated area



Xilinx XPower Analyzer - AGING_AWARE_RELIABLE_MULTIPLIER.ncd - [Table View]

Device	
Family	Spartan3e
Part	xc3s1600e
Package	fg320
Temp Grade	Commercial
Process	Typical
Speed Grade	-4

On-Chip	Power (W)	Used	Available	Utilization (%)
Clocks	0.000	3	---	---
Logic	0.000	21	29504	0
Signals	0.000	37	---	---
IOs	0.000	22	250	9
Leakage	0.203			
Total	0.203			

Supply Summary		Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	1.200	0.069	0.000	0.069
Vccaux	2.500	0.045	0.000	0.045
Vcco25	2.500	0.003	0.000	0.003

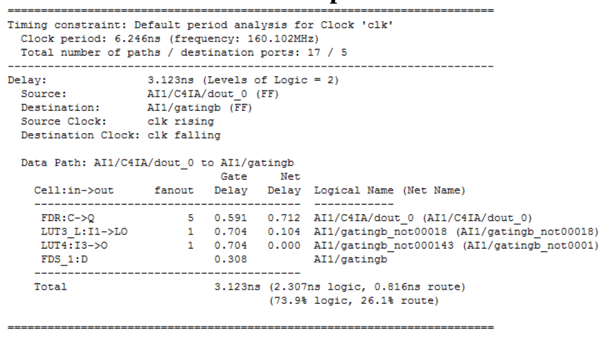
	Total	Dynamic	Quiescent
Supply Power (W)	0.203	0.000	0.203

Environment	
Ambient Temp (C)	25.0
Use custom TJA?	No
Custom TJA (C/W)	NA
Airflow (LFM)	0

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	21.1	80.7	29.3

Characterization	
PRODUCTION	v1.2,06-23-09

Tabulated power



```
================================================================================
Timing constraint: Default period analysis for Clock 'clk'
  Clock period: 6.246ns (frequency: 160.102MHz)
  Total number of paths / destination ports: 17 / 5
--------------------------------------------------------------------------------
Delay:               3.123ns (Levels of Logic = 2)
  Source:            AI1/C4IA/dout_0 (FF)
  Destination:       AI1/gatingb (FF)
  Source Clock:      clk rising
  Destination Clock: clk falling

  Data Path: AI1/C4IA/dout_0 to AI1/gatingb
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDR:C->Q              5   0.591   0.712  AI1/C4IA/dout_0 (AI1/C4IA/dout_0)
     LUT3_L:I1->LO         1   0.704   0.104  AI1/gatingb_not00018 (AI1/gatingb_not00018)
     LUT4:I3->O            1   0.704   0.000  AI1/gatingb_not000143 (AI1/gatingb_not0001)
     FDS_1:D                   0.308          AI1/gatingb
    ----------------------------------------
    Total                      3.123ns (2.307ns logic, 0.816ns route)
                                       (73.9% logic, 26.1% route)

================================================================================
```

Tabulated delay

References

[1] B. C. Paul, H. Kufluoglu, K. Kang, M. A. Alam, and K. Roy, "Impact of NBTI on the temporal performance degradation of digital circuits," IEEE Electron Device Lett., vol. 26, Aug. 2005.

[2] A. K. Verma, P. Brisk, and P. Ienne, "Variable latency speculative addition: A new paradigm for arithmetic circuit design," in Proc. DATE, 2008.

[3] H.-I. Yang, S.-C. Yang, W. Hwang, and C.-T. Chuang, "Impacts of NBTI/PBTI on timing control circuits and degradation tolerant design in nanoscale CMOS SRAM," IEEE Trans. Circuit Syst., vol. 58, Jun. 2011

[4] Ing-Chao Lin, Yi-Ming Yang and Yu-Hung Cho, "Aging Aware Reliable Multiplier Design With Adaptive Hold Logic" IEEE Transaction on VLSI systems, vol. 23, no. 3, March 2015.

[5] Dr.Vimala Palanichamy and Swapna M ,"Delay Comparison of Various Multipliers With Adaptive Hold Logic(AHL)"- June 2015.

[6] J. Ohban, K. Inoue and V. G. Moshnyaga, "Multiplier energy reduction through bypassing of partial products," in Proc. APCCAS, 2002.

[7] J. S. Chitode, Neha V. Mahajan, "Simple Computation of DIT FFT" in IJARCSSE, Volume 4, Issue 5, May 2014.

[8] M.-C. Wen, S. J. Wang, and Y.-N. Lin, "Low power parallel multiplier with column by passing," in Proc. IEEE ISCAS, May 2005

Author Profile

S. Arthi, received B.E degree in Electronics and Communication Engineering in the year 2015 and Pursuing M.E in VLSI Design at Karpaga Vinayaga College of Engineering and Technology affiliated to Anna University, Chennai-India.

G. Prathipa, Assistant professor, Department of Electronics and communication engineering, Karpaga Vinayaga College of Engineering and Technology, Chennai- India.