VLSI Implementation of VOLDER'S Algorithm in OFDM for various Applications

Dauzi Imran Khan, Dr. Shaik Saheb Basha

Department Of ECE ,
Madina Engineering College Kadapa ,
JNTUA University, A.P., India

Abstract

VOLDER revolves around the idea of "rotating" the phase of a complex number, by multiplying it by a succession of constant values. However, the "multiplies" can all be powers of 2, so in binary arithmetic they can be done using just shifts and adds, no actual "multiplier This paper deals with implementation of the vlsi synchronizer and VOLDER algorithm for OFDM based applications. The main principle of VOLDER are calculations based on shift-registers and adders instead of multiplications, what saves much hardware resources. VOLDER is used for polar to rectangular and rectangular to polar conversions and also for calculation of trigonometric functions, vector magnitude and in some transformations, like discrete Fourier transform (DFT) or discrete cosine transform (DCT). So the primary task is to understand the algorithm and then create a VHDL description. VOLDER is a clear winner when a hardware multiplier is unavailable,. The different computations were carried out by giving appropriate test cases and the results were verified with the help of its simulation waveforms in Modelsim tool. Also the design can be synthesized using the Xilinx tool.

Keywords: *Synchronizer, VOLDER, OFDM, FPGA.*

I. Introduction

DSP applications impose several challenges on the implementation of the DSP systems. These implementations must satisfy the enforced sampling rate constraints of the real time DSP applications and must require less space and power consumption, At present, digital signal processing (DSP) has always been driven by the advances in DSP applications and in scaled very-large-scale-integration (VLSI) technologies. Therefore, at any given time,. The flexibility, high throughput, low power and low cost are driving forces for developing VLSI (FPGA/ASIC) based design of present mobile and wireless communications.

The property of orthogonality allows simultaneous transmission on a lot of sub-carriers in a tight frequency space without interference from each other. OFDM abbreviated as orthogonal frequency division multiplexing. It is a combination of modulation and multiplexing Independent signals that are a sub-set of a main signal are multiplexed in OFDM and also the signal itself is first split into independent channels, modulated by data and then re-multiplexed to create the OFDM carrier.

Orthogonality of the sub-carriers is the main concept in OFDM. In recent years the telecommunications industry has experiencing a tremendous growth in the area of wireless communication. This growth has ignited the wide spread popularity of mobile phones and wireless computer networking. The growth in multi input and multi output systems increase demand for fast communications with in an allotted bandwidth resources [1], [2] , [3], [4].

II. FOCUSING CONSEQUENCES

This paper discusses to implement a single input and single output system without interference of channel noise in OFDM based connections. The area of wireless communication combined with VLSI system design has drawn my attention. The OFDM is the next generation technology, and deals with two small and main components namely autocorrelation and synchronization of the received signals and VOLDER for computation of channel estimation and its compensation of OFDM in the wireless receiver. The VLSI implementation of Synchronizer and pipelined VOLDER in OFDM receiver implemented using 130 nm technologies.

This paper discusses and classified as follows. The first section gives the introduction, the second section states the problem, third section gives the design aspects of synchronizer and fast pipelined VOLDER. The Section four shows the VLSI implementation aspects of both autocorrelator and VOLDER and finally concluded in last section.

III. DESING CONSIDERATIONS

A. Synchronizer

Autocorrelation is also representing as the cross-correlation of a signal with itself. Autocorrelation is useful for finding repeating patterns in a signal [5]. Synchronizer is an autocorrelator. Autocorrelation is a measure of how well a signal matches a time shifted version of itself, as a function of the amount of time shift. It is best suitable for wireless applications for transmitting and receiving signal. Autocorrelation involves only one signal and provides

It determines the presence of a periodic signal which information about the structure of the signal or its behavior in the time domain. The autocorrelator takes care of frame synchronization, time synchronization, frequency synchronization and carrier frequency offset estimation by cross-correlating the received signal with a delayed version of itself[6]. The implementation of the autocorrelator in this design requires one eight bit multiplier, an adder and a 128-by-8 bit wide RAM. An autocorrelator which can take in input of length 128 samples each of 8 bits wide is designed. Fig.1 shows the block diagram of the synchronizer. The design is carried out in Verilog HDL and simulated using Model Sim. The autocorrelator is designed in such a way that 'N' number of inputs can be fed in and the output port named as 'line' displays the auto correlated version of it. In this design a RAM was modeled to accommodate 128 samples each of 8 bits wide. The address of the RAM is generated with the help of a counter and the outputs of the RAM are 'a' and 'b'. For example if the address of 'a' is [addr], then address of 'b' is minus one position i.e. [addr-1]. Therefore when the first sample goes into the RAM, it is stored in 'a' register and 'b' is 0. When the second sample goes into the RAM, the previous value of the register 'a' is stored in 'b' and 'a' gets the new value. This operation in other terms can be called as the shift operation. Here the each sample is getting shifted by one position i.e. 8 bits. The writing operation is done only when the write enable pin (we) is high and the 'we' becomes low when the RAM is filled. Reset pin is turned high for 10ns after which it is made low. The duty cycle of the clock is 100ns. The writing operation occurs in the negative or falling edge of the clock to save the writing time. Therefore, for 128 samples, this design performs 127 forward shifts and produces 127 outputs serially from a single output port namely 'line'. The accumulator accumulates the result and is fed from an intermediate register only in the event of a clock.

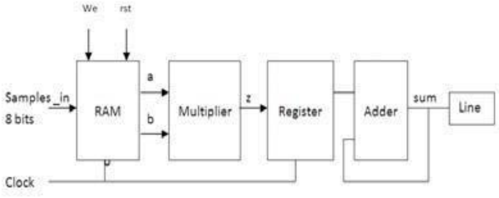

Figure. 1 Black diagram of the Synchronizer

$$r(j) = \frac{1}{N}\sum_{n=0}^{N-1-|j|} x(n)x(n+j)$$

The first section gives the introduction, the second section states the problem, third section gives the design aspects of synchronizer and fast pipelined VOLDER.

The Section four shows the VLSI implementation aspects of both autocorrelator and VOLDER and finally concluded in last section.

The design is carried out in Verilog HDL and simulated using Model Sim.

TABLE.I DESIGN PARAMETERSS OF SYNCHRONIZER ON THREE DIFFERENT XILINXS FPGA TARGETS

Parametr	Spartan 2	Spartan 3E	Vertex 2Pro
Maximum Operating Frequency (MHz)	50.241	108.221	114.2
Total Estimation Power consumption (mW)	3.5	17	22.1
Total Minimum Time Period (ns)	4.55	1.22	2.130
Memory Usage (K Bytes)	63802	74584	82554

B. VOLDER Algorithm PIPELINED

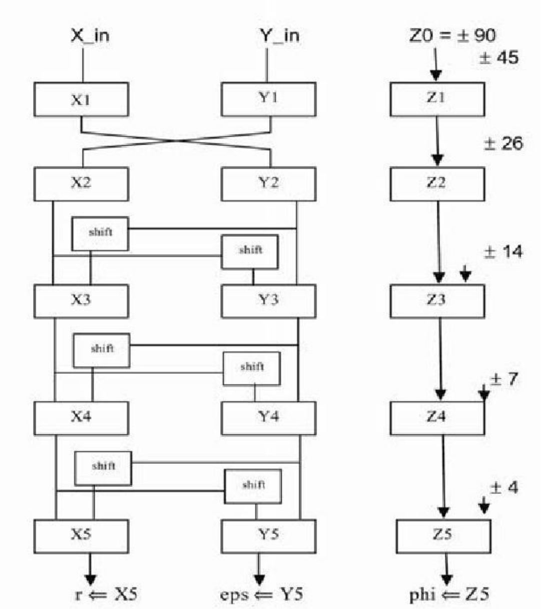

Figure 2: Block Diagram of fast pipelined VOLDER

VOLDER stands for Rotation DIgital Computer Algorithm. VOLDER is a simple and efficient algorithm to calculate hyperbolic and trigonometric functions. The VOLDER algorithm provides an iterative method of performing vector rotations by arbitrary angles using only shifts and adds. The algorithm, credited to Volder [7] is derived from the general (Givens) rotation transform.

IV. ARC TANGENT

The arc tangent o= Atan (Y/X) is directly computed using the vectoring node Vector rotator if the angular accumulator is initialized with zero . The argument must be provided using the ratio expressed as vector

(X,Y).Presenting the argument as a ratio has the advantage of being able to represent the infinity (by setting X=0).since the arctangent result is taken from the angle accumulator, the VOLDER Rotator growth does not affect the result.

Zn= Zo+ TAN^-1(Y0/X0).

V. VECTOR MAGNITUDE

The vectoring node VOLDER rotator reduces the magnitude of the input vector as byproduct of computing the arctangent .After the vectoring mode rotation, the vector is aligned with the X- axis .The magnitude of the vector is therefore the same as the X component of the rotated vector. This result is apparent in the result equations for rotator mode rotator

Xn= An ^(XO^2 + Y0^2)

The magnitude of the result is calculated by the processor again which needs to be accounted for elsewhere in the System .This implementation of vector magnitude has a hardware complexity of one multiplier of the same width. The volder algorithm represents a significant implementation. Over an equivalent pythogorean processor. The accuracy of The magnitude results improves by bits for each iteration performed.

VI. ARC SINE AND ARC COSINE

The arcsine can be computed by starting with a unit vector on the positive x axis, then rotating it so that its y component is equal to the input argument. The arcsine is then the angle subtended to cause the y component of the rotated vector to match the argument. The decision function in this case is the result of a comparison between the input value and the y component of the rotated vector at each iteration:

Xi+1 = Xi+Yi..di.2^-i

Yi+1 = Yi+Xi.di.2^-i

Zi+1 =Zi-di.tanh^-1(2^_i)

Where di =-1 if Zi<C , - 1 otherwise .

C= INUT ARGUMENT

Rotation produces the following results

Xn= ^(An.xo^)2 – c^2

Yn= C

Zn= Z0+ Arcsin (C/AO.Xn)

An=TI n 1+ 2i

The arcsine functions as stated above returns correct angles

For inputs -1< C/AO.Xn <-1. Although the accuracy affects the inut approaches +- 1. (The error increases rapidly for the inputs greater than 0.98).This losss of the accuracy is due to the gain of the rotator. For angles near the Y-Axis, the rotator gain causes the rotated vector to be shorter than the reference (input values). So the decisions are taken improperly. The gain problems can be solved by using the double iteration Algorithm at the cost of increasing in the complexity.

The Arccosine computation is similar, except the difference between x component and the input is used as the decision function. Without modifications, the arccosine algorithm works only for inputs less than making the double iteration algorithm a necessity. The Arccosine could also be computed by using arcsine function subtracting from the result, followed by an angular reduction if the result is in the fourth quadrant.

The autocorrelator is designed in such a way that 'N' number of inputs can be fed in and the output port named as 'line' displays the auto correlated version of it. In this design a RAM was modeled to accommodate 128 samples each of 8 bits wide. First a brief description of the theory behind the algorithm and the derivation of several functions is presented. Then the theory is extended to the so-called unified Volder algorithm. The advent of reconfigurable logic computers permits the higher speeds of dedicated hardware solutions at costs that are competitive with the traditional software approach.

Verilog HDL is written to realize a fast pipelined VOLDER processor and an exhaustive test bench is also written to simulate the functionality with the help of ModelSim. by the tangent term is reduced to simple shift operation. Fig.2 shows a pipelined VOLDER processor which employs iterative VOLDER architecture. It can be obtained simply by duplicating each of the three difference equations shown in equations (2) to (8). The decision function, di, is driven by the sign of the 'Y' or 'Z'register depending on whether it is operated in rotation or vectoring mode. In operation, the initial values are loaded into the 'X' and 'Y' registers. Then on each of the next 'n' clock cycles, the values from the registers are passed through the shifters and adder-subtractors and the result placed back in the registers. The shifters are modified on each iteration to cause the desired shift for the iteration. The divide operation is performed with the help of right shift operation. For divide by 2 operations, the MSB is retained and the remaining 7 bits are right shifted. For the next condition when a divide by 4 operations is required, the 2 MSB's are retained and the remaining 6 bits are right shifted. As the iterations increase, we get an exact value and an exhaustive test bench is also written to simulate the functionality with the help of ModelSim.

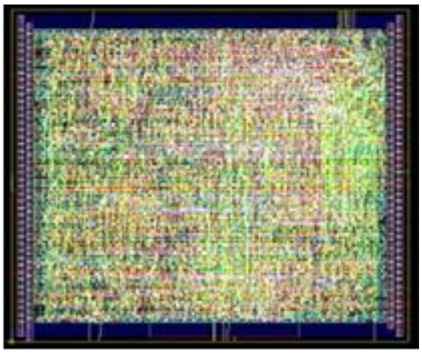

Figure. 3 Chip layout of OFDM Synchronizer

TABLE II DESIGN PARAMETRES OF PIPELINED VOLDER ON THREE DIFFERENT XILINX FPGA TARGETS

Parameter	Spartan 2	Spartan 3E	Vertex 2Pro
Maximum Operating Frequency (MHz)	6.11	11.422	12.35
Total Estimation Power Consumption (mW)	1.9	18.67	12.12
Total Minimum Time Period (ns)	2.495	1.236	1.99
Memory Usage (K Bytes)	31645	37292	40750

VII. RESULTS OF IMPLMENTATION

A. Synchronizer & Its implementation.

In autocorrelator design, the internal cell power, the net switching power, and the total dynamic power are observed 10.7072 mW (96%), 485.3438 uW (4%) and 11.1925 mW (100%) respectally. The cell leakage power is calculated as 653.0287 mWThe combinational area is occupied in autocorrelator chip is 32377.905020 and non-combinational area is 27367.180801. Finally the total cell area of autocorrelator is 59745.085822. The cell leakage power is calculated as 653.0287 mW. In this design, the data required time is 4.75 ns, the data arrival time is -4.70 ns and the resultant slack is MET with a small negligible variation of 0.05 ns. The design statistics such as number of module cells: 3433; number of pins: 21926; number of IO pins: 37; number of nets: 3517 and average pins per net (Signal): 3.78528. In autocorrelator design, the chip utilization as total standard cell area: 69462.70; core size: width 303.14, height 302.58; area 91724.10; chip size: width 343.14, height 342.58; area 117552.90; cell/core ratio: 75.73%; cell/chip ratio: 59.0906%; number of cell rows: 82. Fig.3 shows the layout of synchronizer [8],[9].

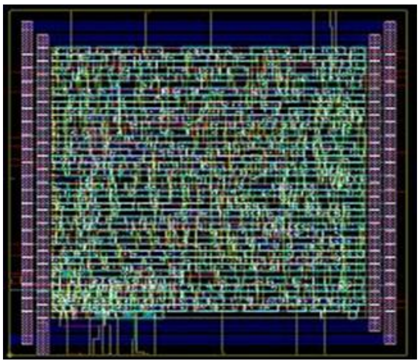

Figure. 4 Chip layout of fast pipelined

B. Pipelined VOLDER

The cell leakage power is calculated as 1nW. The data required time is 4.73ns, the data arrival time is -4.70 ns and resultant slack is MET with a negligible small variation of 0.03 ns. . The combinational area is occupied in VOLDER chip is 6842.219620 and non combinational area is 5284.006336 After performing the physical design, according to area report, the VOLDER design used 41 ports, 555 nets, 390 cells and 32 references.. Finally the total cell area of VOLDER is 12126.225956. In this VOLDER design, the internal cell power, the net switching power, and the total dynamic power are observed 3.2567mW (93%), 227.4490 mW (7%) and 3.4841 mW (100%) respectally.. In VOLDER design, design statistics: number of module cells: 619; number of pins: 3511; number of IO pins: 43; number of nets: 767; average pins per net (signal): 2.7963. In VOLDER design chip utilization: total standard cell area: 12966.44; core size: width 149.96, height 147.60; area 22134.10; chip size: width 189.96, height 187.60; area 35636.50; cell/core ratio: 58.5813%; cell/chip ratio: 36.3853%; number of cell rows: 40. Fig.4 shows the layout of pipelined VOLDER architecture [8],[9].

VIII. CONCLUSION

A fast pipelined VOLDER architecture and auto correlator is designed, implemented and tested. Matlab simulations are for functional verification. This paper deals with VLSI implementation of synchronizer and VOLDER algorithm for OFDM based applications. The function of synchronizer is autocorrelation. The autocorrelator is used for frame detection and carrier frequency offset estimation for fourth generation wireless applications. The VOLDER is effectively used at OFDM receiver to estimate the frequency offset and to calculate the division algorithm for channel estimation. performed prior to the Verilog HDL coding The total cell area of autocorrelation fast pipelined VOLDER is12126.225956. The internal cell power, the net switching power, and the total dynamic power are observed3.2567mW(93%), 227.4490 mW (7%) and 3.4841 mW (100%) respectively.

REFERENCES

[1] RAY ANDRAKA, "A survey of VOLDER algorithms for FPGA based computers".

[2] Sudhakara Reddy.P, Ramachandra Reddy.G, "ASIC Implementation of Autocorrelation and VOLDER Algorithm for OFDM based WLAN", EJSR,Vol.27, No.4, 2009, pp. 588-596.

[3] Meyer-Baese. U, "Digital Signal Processing with Field Programmable Gate Arrays", Springer-Verlag, ISBN 81-8128-045-8, 2003.

[4] Jose.M Canet, Felip Vicedo, Vicenc Almenar, Javier Valls, " FPGA Implementation of an IF transceiver for OFDM-based WLAN", IEEE Workshop on Signal Processing Systems, Vol.2.