OPTIMIZATION AND IMPLEMENTATION OF PARALLEL SQUARER

Sharath D[1], Devaraju G[2], Kavitha Devi C.S[3]

[1]PG Student of VLSI Design and Embedded system in Department of Electronics and Communication, Dr. Ambedkar Institute of Technology, Bangalore
[2]Assoc. Professor in Department of Instrumentation Technology, Dr. Ambedkar Institute of Technology, Bangalore
[3]Asst. Professor in Department of Electronics and Communication Engineering, Dr. Ambedkar Institute of Technology, Bangalore

Abstract

This paper describes the low power architecture design which is implemented in digital systems. It focuses on the design and implementation of Parallel Squarer and its optimization at the gate level netlist. The efficient parallel squarer design is optimized at the gate level and the results are obtained. It's much efficient than the existing system. Parallel squarer architecture is implemented based on Binary squarer algorithm to provide low, power, area and delay with the trade-offs constraints. Simulation, synthesis is done in the Modelsim 6.5 and Xilinx 13.1 v. The gate level netlist simulation and synthesis are done in Cadence Encounter RTL compiler and the results are compared with Xilinx and cadence tool.

Keywords*- Binary Squarer, Cadence, Modelsim, Parallel Squarer, etc*

---***---

1. INTRODUCTION

In the advance VLSI technology, low power architecture implementations for the better performance and for the power resources are an important constraint. In this paper a proposed Low power architecture Design has been constructed by the design of Parallel Squarer using Binary squarer algorithm. Optimized parallel squarer is designed using array multiplier and carry look ahead adder and the efficient results are obtained by the optimization of the proposed design. Section II describes the squarer, parallel squarer and the block diagram associated with it and the components in it. Section III describes the binary squarer diagram which is designed and with examples. Section IV describes the implementation and the optimized circuits of the proposed design. Section V describes the simulation and synthesis results using the cadence 180nm technology encounter RTL compiler. Section VI gives the future scope, Section VII provides the conclusion followed by reference. The hybrid squarer and the combined squarer are the future works of the proposed architecture.

2. SQUARER

Squarer is a digital multiplier with the identical inputs .It consists of multiplier unit which is designed for the DSP applications with various purposes in Viterbi decoding, image compression, Euclidean distance and vector normalization, pattern recognition, adaptive filtering, graphic processors, digital synthesizers, polynomial and filter evaluation and soon. Squarer can be designed using binary unsigned and signed numbers. In this paper a unsigned binary squarer is been designed and implemented in the digital systems. Fig. 1 shows the block diagram of a squarer circuit.

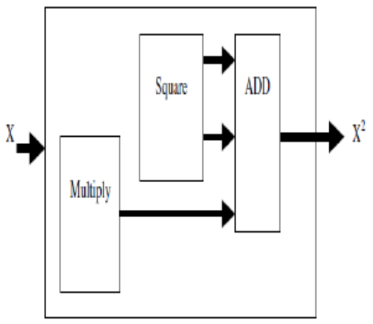


Fig.1 Block diagram of squarer circuit[1]

There are 3 units used: Multiply, Square and Add. Multiply unit contains the multiplier and multiplicand designed using any multiplier, Square unit is a multiplier with identical inputs .Add unit contains the adder for adding the products of multiply and square unit.

2.1 Parallel Squarer

Parallel Squarer is a squaring circuit where the squaring operations using multiplier and the addition of products take place parallelly in a digital system. Parallel Squarer is a binary squarer where the multiplier is computed parallel with the use of array multipliers to yield the products. The block diagram of parallel squarer is shown in Fig.2.Parallel squarer reduces the computation time by reducing the execution cycles by parallel computing the result of squaring unit. There are three steps in obtaining the squared number: 1.Partial Product Generation (PPG), 2.Partial Product Reduction (PPR) 3.Partial Product Addition (PPA).First step is done using the array multiplier to generate the products in the proposed system. Second step is the optimization of the product by reduction in the structural gates. Third step is the addition of the optimized result. By these steps a efficient parallel squarer of low power architecture can be obtained. In proposed system, Parallel squarer uses array multiplier with identical inputs for squaring circuit, CLA for addition. Other multipliers and adders can be used for the computation. Array multiplier is used as it takes up least amount of area and less wiring and routing in layout. CLA reduces the propagation delay. Array multiplier has regular structure.

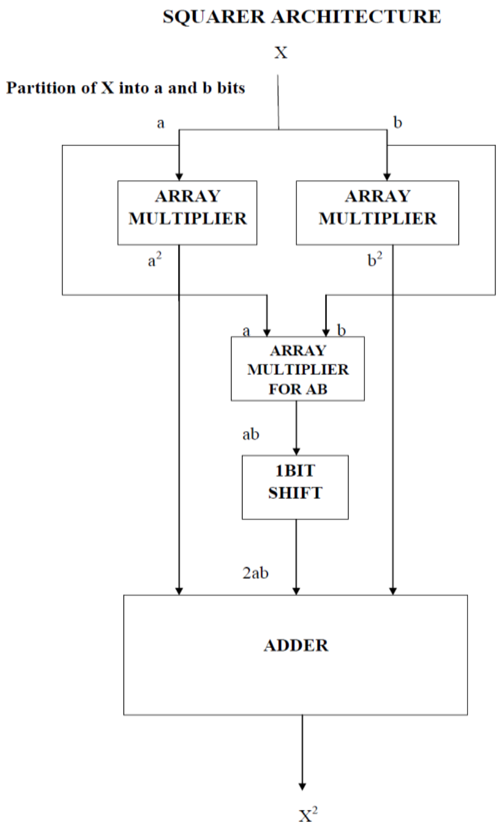


Fig 2 Block diagram of Parallel Squarer

2.2 Array Multiplier

Array multiplier uses AND gates for generation of the bit-products and adders for accumulation of generated bit products. All bit-products are generated in parallel and collected through an array of full adders and half adders. Since the array multiplier is having a regular structure, wiring and the layout are done in a much simplified manner [2].

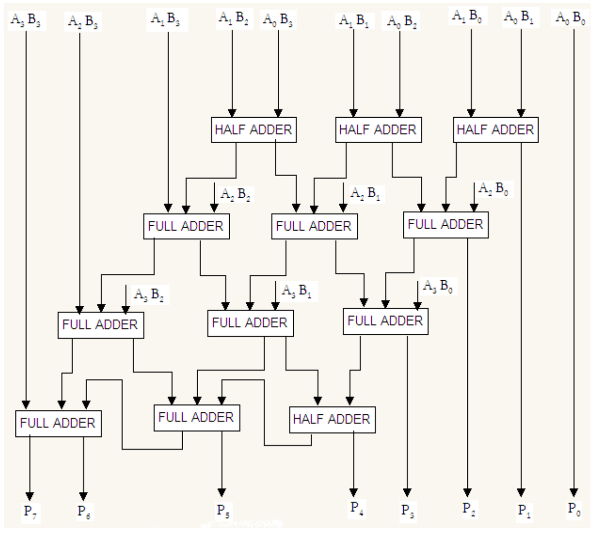


Fig 3: Array Multiplier using CSA Hardware Architecture[2]

2.3 Carry Look-Ahead Adder (CLA)

Carry Look-ahead adder (CLA) is used for the addition of the products obtained in Parallel squarer. The addition of a2,b2 and 2ab takes place in CLA. The block diagram of 8 bit CLA using 4 bit CLA is shown in Fig. 4. In our proposed architecture a 8 bit CLA is used using a 4 bit CLA. The 4 bit squarer and a 8 bit squarer uses a 8 bit CLA and two 8 bit CLAs.

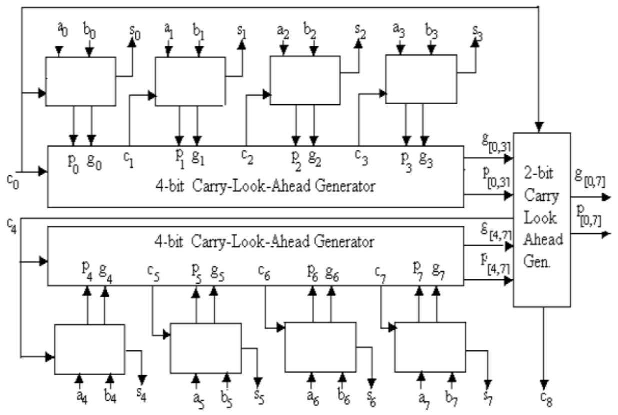

4-bit Carry-Look-Ahead Generator
4-bit Carry-Look-Ahead Generator
2-bit Carry Look Ahead Gen.
c0
c4
c8

Fig .4 Block diagram of 8-bit Carry Look ahead adder using 4-bit CLA

3. BINARY SQUARER ALGORITHM

Squares are a special case of multiplication where both inputs are identical.

Proposed Algorithm:

The algorithm consists of following steps:
The given input is partitioned into two parts, each part is treated as a separate unit processed individually by further units.

- Find the square of each part.
- Find twice the product of individual part.
- Add the above results suitably to get the final result.

If X is a five-digit number, who’s square has to be computed.

X = abcde.

Find square of abc = $(abc)^2$.
Find square of de = $(de)^2$.
Find twice the product of abc & de = 2(abc) (de)
Find the sum of the above results to get the square of X.

Example:
1. $(10101111)_2$-8bit binary=$(175)_{10}$
The square of 175 in binary is $(111011110100001)_2$-$(30625)_{10}$

Step 1:
Partition the eight bit binary number to 4 bit binary number (any bit number can be partitioned).

Step 2:
a=$(0110)_2$, b=$(1010)_2$.Perform a-square and b-square by the squaring unit using array multiplier. a-square= $(1100100)_2$-$(100)_{10}$, b-square= $(11100001)_2$-$(225)_{10}$

Step 3:
Perform 2ab by multiplying a and b to get the products ab then shift left to get 2ab, ab= $(10010110)_2$, 2ab= $(100101100)_2$

Step 4:
The MSB bit partitioned b-square numbers is retained and added with 2ab and a-square to get square of the number. The bolded numbers is b-square. Shift left by 4 for 2ab and add a-square and b-square to obtain the square of the given number.

00**100101100**0000
+ **1100100****11100001**
111011110100001

Bold Blue indicates 2ab value, Red Bold indicates a^2, and Black bold indicates b^2 value.

4. IMPLEMENTATION DESIGN

4.1 4-Bit Squarer

The implementation is done using the structural gate levels using the cadence tool. The unoptimised circuits and optimised circuits are shown in Fig. 5.

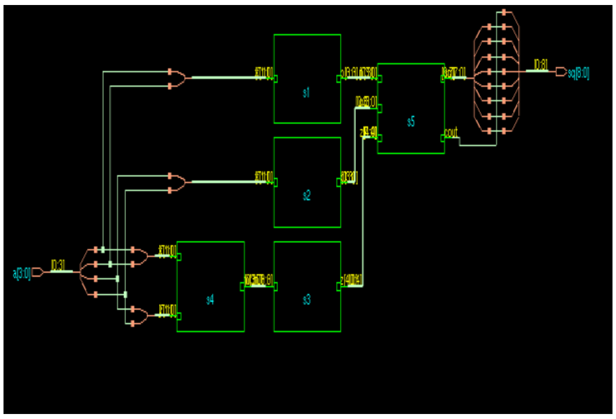
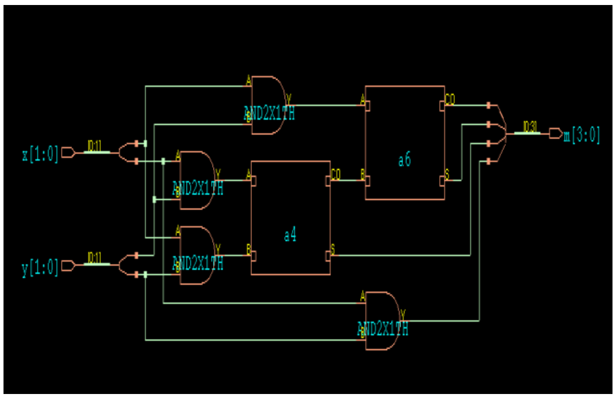
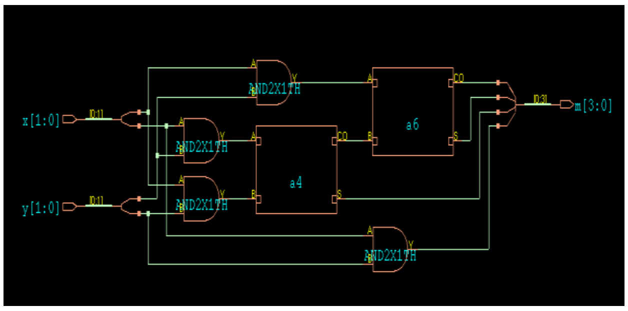


Fig. 5 Block diagram of 4-bit squarer in cadence (unoptimised circuit i.e., existing system).

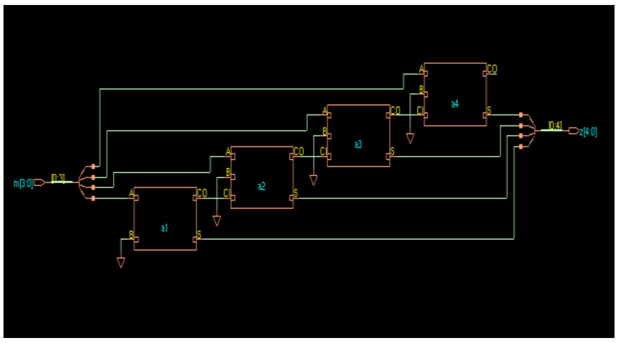
In the existing system 5 component blocks are used and the sub blocks are shown in Fig 6



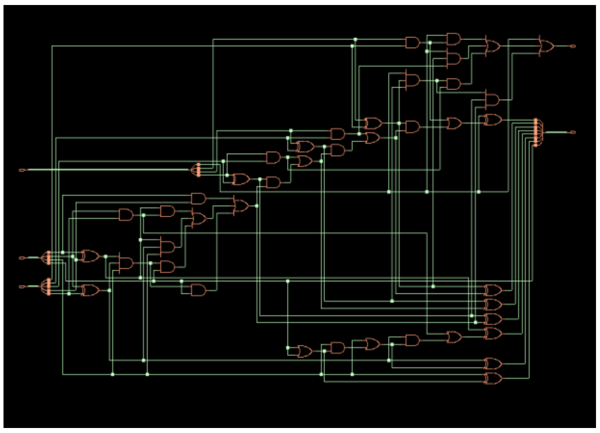
(a)



(b)



(c)



(d)

Fig. 6 Components of unoptimised 4-bit squarer (a)X^2 or Y^2 block (b)XY block (c)2XY blockd (d)CLA_8bit block

The components of squarer are shown in diagram 5 blocks are used in unoptimised or existing system architecture which has more delay and power.The optimised squarer and the components are shown in Fig. 7 and Fig. 8.

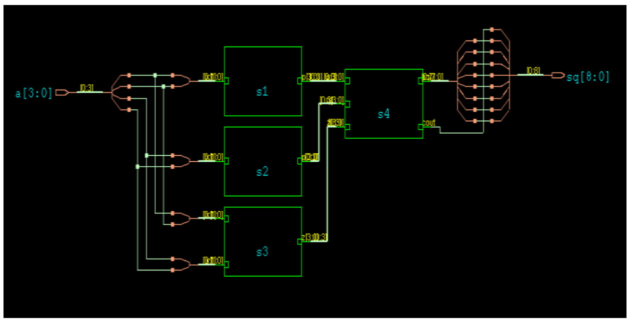
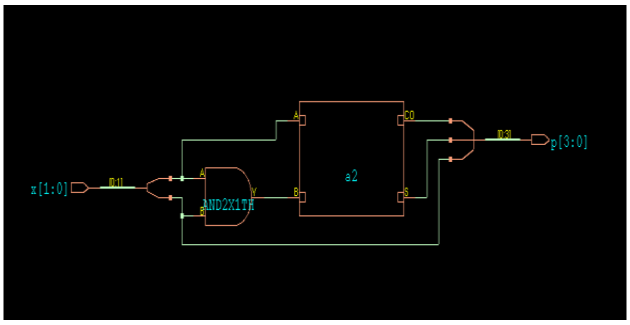
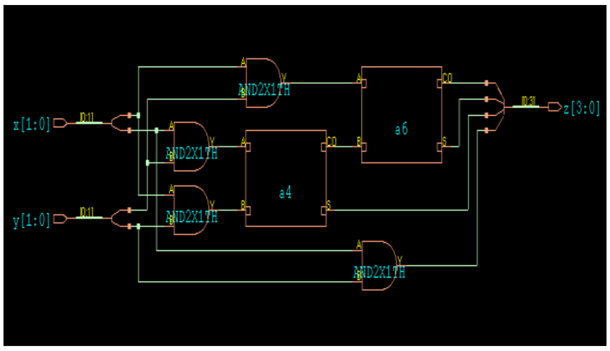


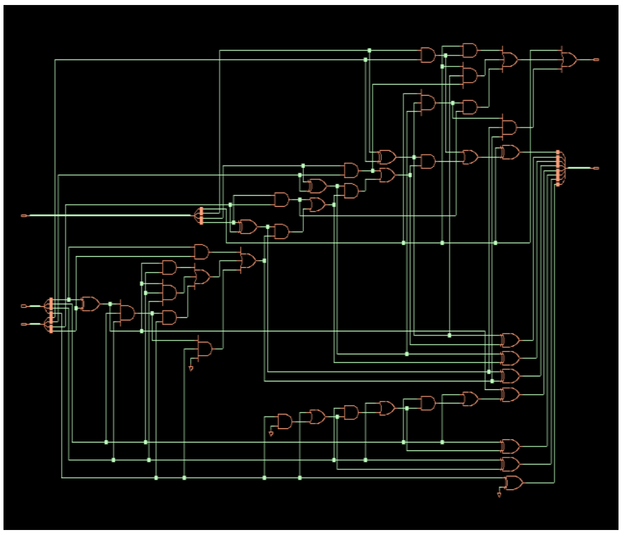
Fig. 7 Block diagram of Optimised 4-bit squarer in cadence



(a)



(b)



(c)

Fig. 8 Components of 4-bit optimised squarer (a)X^2 or Y^2 block(b)XY block(c)CLA_8bit block

We can notice that the components of optimised parallel squarer is reduced and 2XY block is removed.The optimisation is such a way that the adder appends zero bit to MSB of XY while adding so that 2XY is added and the result is obtained.The same procedure follows for 8 bit squarer also.

4.2 8-Bit Squarer

8-bit squarer is designed using the 4-bit squarer is used for X^2block and 2XY block of 4-bit squarer is used for 2XY block too in 8-bit squarer. Totally, 8-bit Squarer is designed using4 bit squarer. The unoptimised and optimised squarers are shown in Fig. 9 and Fig. 10.

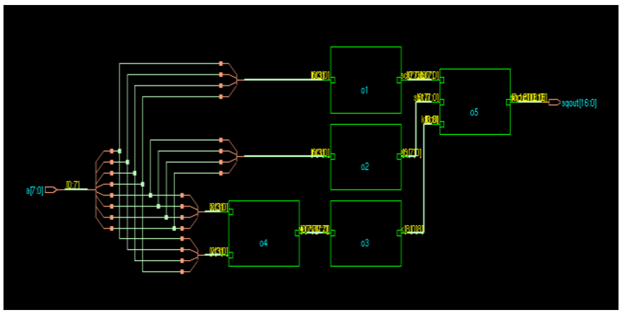
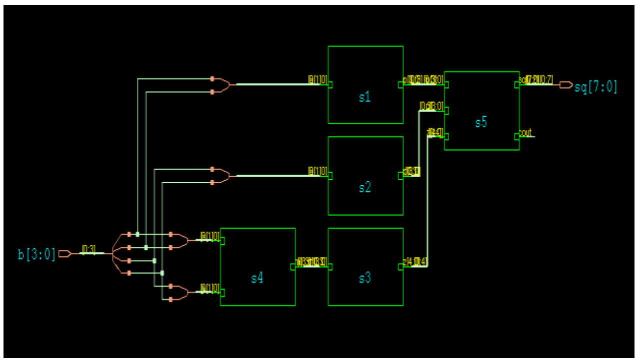
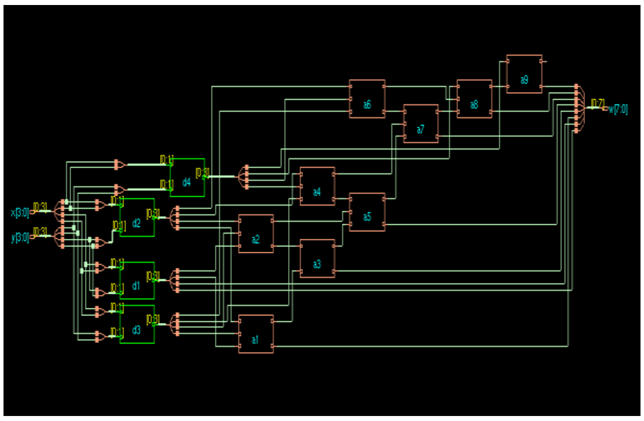


Fig. 9 Block diagram of 8-bit squarer in cadence (unoptimised circuit i.e., existing system).

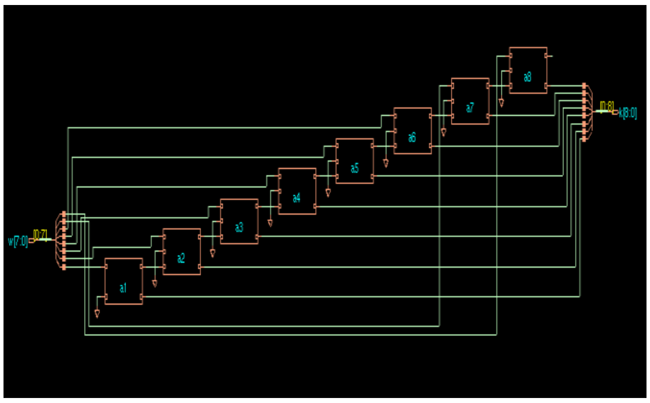
In the existing system 5 component blocks are used and the sub blocks are shown in Fig. 10.



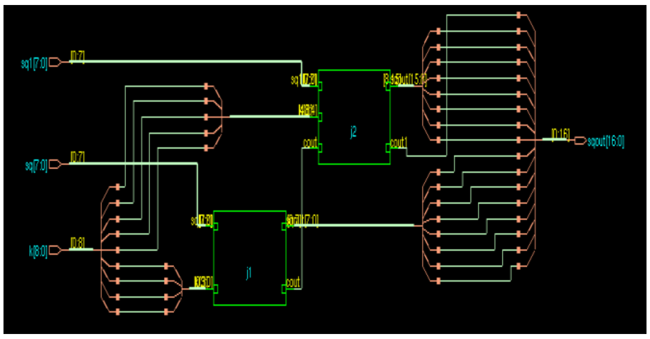
(a)



(b)



(c)



(d)

Fig. 10 Components of unoptimised 8-bit squarer (a)X^2 or Y^2 block (b)XY block (c)2XY blockd (d)Adder block

The components of squarer are shown in diagram 5 blocks are used in unoptimised or existing system architecture which has more delay and power.The optimised squarer and the components are shown in Fig. 11 and Fig. 12.

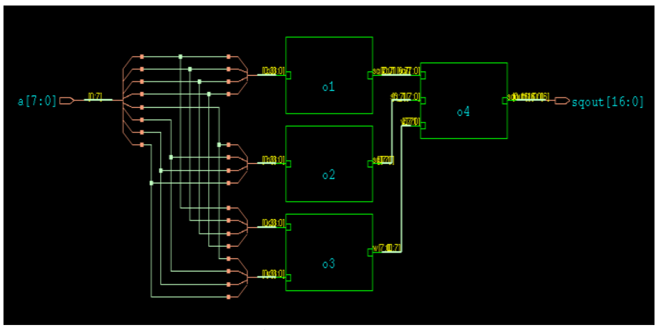
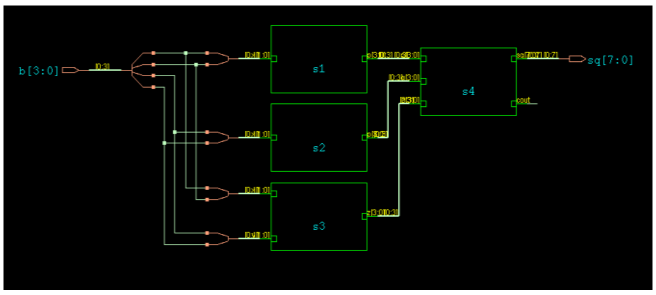
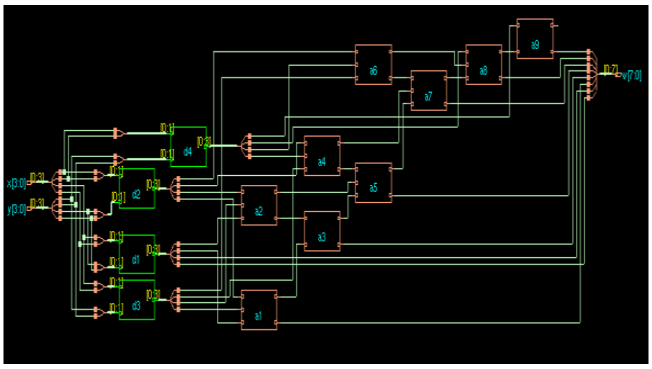


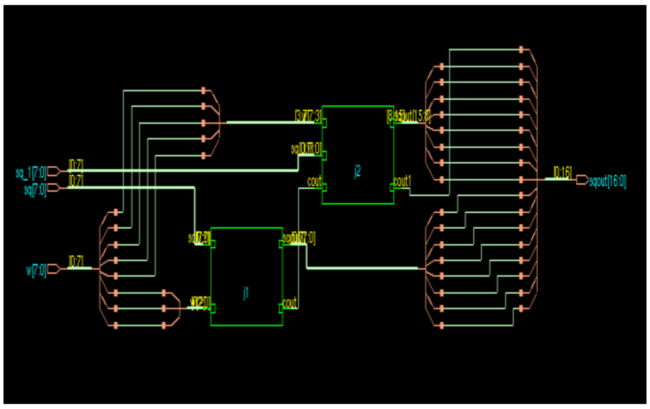
Fig. 11 Block diagram of Optimised 8-bit squarer in cadence



(a)



(b)



(c)

Fig. 12 Components of optimised 8-bit squarer (a)X^2 or Y^2 block (b)XY block (c)Adder block

5. SIMULATION AND SYNTHESIS RESULTS

The simulation is done using the MODELSIM 6.5 version for 4 bit and 8 bit squarers, and report of area, timing, power is obtained in Cadence Encounter(R) RTL compiler version(RC)12.10.

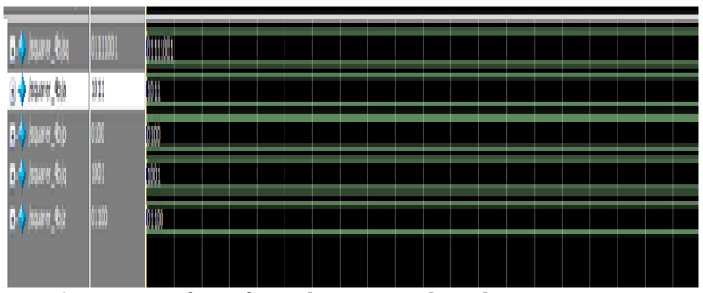


Fig 13 Waveform from the RTL code -4 bit squarer

In the above simulation a=1011 i.e. 11 in decimal and the squared output sq=001111001 is obtained.

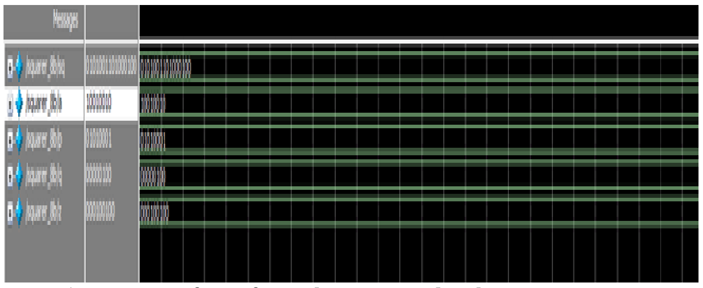


Fig 14 Waveform from the RTL code-8bit squarer

In the above simulation a=10010010 i.e. 146 in decimal and the squared output sq=101001101000100 i.e., 21316 is obtained. Comparison of Power, Area and Timing of unoptimised and optimised Squarers of 4 and 8 bit elaborated in Cadence RTL compiler and the improved performance based on the parameters are tabulated.

Table 1 Power, Area and Timing of 4-bit squarer

PARAMETER-S	4-BIT SQUAR-ER(EXI-STING)	4-BIT SQUA-RER (PROPOSED)	REDUCE PERCENTAGE (%)
TIMING(pS)	2490 F	2164 F	13.09
AREA(CELLS)	64	52	18.75
LEAKAGE POWER(nW)	316.243	209.213	33.84
DYNAMIC POWER(nW)	3626.517	2464.238	32.04
TOTAL POWER(nW)	3942.760	2673.451	32.19

Table 2 Power, Area and Timing of 4-bit squarer

PARAMETERS	8-BIT SQUARER (EXISTING)	8-BIT SQUARER (PROPOSED)	REDUCE PERCENTAGE (%)
TIMING(pS)	5001F	4814 F	3.73
AREA(CELLS)	253	217	14.22
LEAKAGE POWER(nW)	1397.870	999.021	28.53
DYNAMIC POWER(nW)	23582.993	18355.838	22.16
TOTAL POWER(nW)	24980.863	19354.858	22.52

6. FUTURE ENHANCEMENT

Optimized Parallel squarer can be further optimized at the gate and structural level and can be combined with signed and unsigned multipliers to obtain hybrid squarer. Hybrid squarer provides the performance better than parallel and binary squarer. Other multipliers can be used for squaring and multiplication unit to provide efficient power. There will be a trade-off between power, area and delay, even with the performance of the circuit in digital system.

7. CONCLUSIONS

An Optimized Parallel Squarer is designed and Implemented for Low power architecture in various DSP applications using the binary squarer algorithm and found that it's efficient in terms of area, power and delay concerned when compared to the existing system. The Parallel Squarer is simulated, optimized and synthesized to get the report of Power, Area and Delay in RTL compiler of Cadence. From the result we can note that an efficient parallel squarer is designed which is optimized based on area and power is reduced when compared to unoptimised architecture.

REFERENCES

[1]. Manasa S.N, S.L.Pinjare, Chandra Mohan Umapthy,"PARALLEL SQUARER DESIGN USING PRE-CALCULATED SUM OF PARTIAL PRODUCTS", *International Journal of Emerging Technology and Advanced Engineering* Website: www.ijetae.com (ISSN 2250-2459, Volume 2, Issue 7, July 2012).

[2]. L. Sriraman, T.N. Prabakar," FPGA IMPLEMENTATION OF HIGH PERFORMANCE MULTIPLIER USING SQUARER", *Int. Jr. of Advanced Computer Engineering & Architecture* Vol. 2 No. 2 (June-December, 2012).

[3]. Hassan Ali and T. Nandha Kumar,"ON THE IMPROVED IMPLEMENTATIONS OF PRE CALCULATED SUMS OF PARTIAL PRODUCTS BASED 7-BIT UNSIGNED PARALLEL SQUARER,"*2013 26th IEEE Canadian Conference Of Electrical And Computer Engineering (CCECE).*

[4]. K.-J. Cho and J.-G. Chung,”Parallel squarer design using pre-calculated sums of partial products”,*ELECTRONICS LETTERS* 6th December 2007 Vol. 43 No. 25.
5. Eshraghi, A., Fiez, T.S., Winters, K.D., and Fischer, T.R.: ‘Design of a new squaring function for the Viterbi algorithm’, *IEEE J. Solid-State Circuits*, 1994, 29, (9), pp. 1102–11072.
[6]. Yoo, J.T., Smith, K.F., and Gopalakrishnan, G.: ‘A fast parallel squarer based on divided-and-conquer’, *IEEE J Solid-State Circuits*, 1997, 32,(6), pp. 909–912.
[7]. Chen, T.C.: ‘A binary multiplication scheme based on squaring’, *IEEE Trans. Comput.*, 1971, 20, (4), pp. 678–680
[8]. Kolagotla, R.K., Griesbach, W.R., and Srinivas, H.R.: ‘VLSI implementation of 350 MHz 0.35 mm 8 bit merged squarer’, *Electron. Lett.*, 1998, 34, (1), pp. 47–48.