IOSR Journal of Engineering (IOSRJEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
Vol. 15, Issue 4, April 2025, ||Series -1|| PP 38-45

FPGA Implementation of Low Power Signed Multiplier for DSP Applications

D.SAI SURYAVATHI[1], CH.SAI SUSHMA[2], P.DURGA BABU[3], K.PRASANNA MALLIKARJUNA[4], K. HIMA BINDU[5]

[1,2,3,4] *B.Tech Scholars, Department of ECE,*
[5]*Assistant Professor, Department of ECE, Aditya College of Engineering and Technology, Surampalem, Kakinada, AP, India.*

ABSTRACT
Multiplication is one of the most commonly used arithmetic operations in a wide range of applications, such as multimedia processing and artificial neural networks. Multiplier is one of the main causes of energy consumption, critical path latency, and resource utilization for these kinds of applications. In designs based on field-programmable gate arrays (FPGAs), these effects become more noticeable. Still, the majority of cutting edge designs were created for ASIC-based systems. Moreover the majority of existing field programmable gate array (FPGA) designs are restricted to unsigned integers and necessitate additional circuits to accommodate signed operations.
To address these limitations for the FPGA based implementations of applications utilizing signed numbers, this letter presents an area- optimized, low-latency, and energy efficient design for an accurate signed multiplier.
KEYWORDS: Multiplier, Energy consumption, Critical path latency, Multimedia processing, Field-programmable gate arrays (FPGAs), ASIC-based systems, Signed numbers, Area-optimized design, Low-latency design, Critical path latency.

I. INTRODUCTION

Modern consumer electronics make extensive use of Digital Signal Processing (DSP) providing custom accelerators for the domains of multimedia, communications etc. Typical DSP applications carry out a large number of arithmetic operations as their implementation is based on computationally intensive kernels, such as Fast Fourier Transform (FFT), Discrete Cosine Transform (DCT), Finite Impulse Response (FIR) filters and signals' convolution. As expected, the performance1 of DSP systems is inherently affected by decisions on their design regarding the allocation and the architecture of arithmetic units.

Recent research activities in the field of arithmetic optimization [1], [2] have shown that the design of arithmetic components combining operations which share data, can lead to significant performance improvements. Based on the observation that an addition can often be subsequent to a multiplication (e.g., in symmetric FIR filters), the Multiply-Accumulator (MAC) and Multiply-Add (MAD) units were introduced [3] leading to more efficient implementations of DSP algorithms compared to the conventional ones, which use only primitive resources [4]. Several architectures have been proposed to optimize the performance of the MAC operation in terms of area occupation, critical path delay or power consumption [5]–[7]. As noted in [8], MAC components increase the flexibility of DSP datapath synthesis as a large set of arithmetic operations can be efficiently mapped onto them. Except the MAC/MAD operations, many DSP applications are based on Add-Multiply (AM) operations (e.g., FFT algorithm [9]).

The straightforward design of the AM unit, by first allocating an adder and then driving its output to the input of a multiplier, increases significantly both area and critical path delay of the circuit. Targeting an optimized design of AM operators, fusion techniques [10]–[13], [23] are employed based on the direct recoding of the sum of two numbers (equivalently a number in carry-save representation [14]) in its Modified Booth (MB) form [15]. Thus, the carry-propagate (or carry- look-ahead) adder [16] of the conventional AM design is eliminated resulting in considerable gains of performance. Lyu and Matula [10] presented a signed-bit MB recoder which transforms redundant binary inputs to their MB recoding form.

A special expansion of the preprocessing step of the recoder is needed in order to handle operands in carry-save representation. In [12], the author proposes a two-stage recoder which converts a number in carry-save form to its MB representation. The first stage transforms the carry-save form of the input number into signed-digit form which is then recoded in the second stage so that it matches the form that the MB digits request. Recently, the technique of [12] has been used for the design of high performance flexible coprocessor architectures targeting the computationally intensive DSP applications [17]. Zimmermann and Tran [13] present an optimized design of [10] which results in improvements in both area and critical path. In [23], the authors propose the

recoding of a redundant input from its carry-save form to the corresponding borrow-save form keeping the critical path of the multiplication operation fixed.

Although the direct recoding of the sum of two numbers in its MB form leads to a more efficient implementation of the fused Add-Multiply (FAM) unit compared to the conventional one, existing recoding schemes are based on complex manipulations in bit-level, which are implemented by dedicated circuits in gate-level. This work focuses on the efficient design of FAM operators, targeting the optimization of the recoding scheme for direct shaping of the MB form of the sum of two numbers (Sum to MB – S- MB). More specifically, we propose a new recoding technique which decreases the critical path delay and reduces area and power consumption. The proposed S-MB algorithm is structured, simple and can be easily modified in order to be applied either in signed (in 2's complement representation) or unsigned numbers, which comprise of odd or even number of bits. We explore three alternative schemes of the proposed S- MB approach using conventional and signed-bit Full Adders (FAs) and Half Adders (HAs) as building blocks.

We evaluated the performance of the proposed S-MB technique by comparing its three different schemes with the state-ofthe-art recoding techniques [12], [13], [23]. Industrial tools for RTL synthesis [18] and power estimation [19] have been used to provide accurate measurements of area utilization, critical path delay and power dissipation regarding various bit-widths of the input numbers. We show that the adoption of the proposed recoding technique delivers optimized solutions for the FAM design enabling the targeted operator to be timing functional (no timing violations) for a larger range of frequencies. Also, under the same timing constraints, the proposed designs deliver improvements in both area occupation and power consumption, thus outperforming the existing S-MB recoding solutions.

II. Related Work

Modified Booth algorithm has made multiplication easier. It consists of recoding table which has been used to minimize the partial products of multiplier. An adder and the multiplier operator of the unit is combine to form a single add-multiply unit. The fusion of the two operators resulting in Fused Add-Multiply(FAM) operator. In this paper different structured recoding techniques are used to implement the Modified Booth encoder incorporating in FAM. Along with the implementation of recoding techniques, comparison has been done with the existing and the designed Modified Booth recoder.

Electronic world consists of different applicative circuits for particular applications. These circuits comprises of complex arithmetic units. One such field is DSP(Digital Signal Processing) applications such as FFT (Fast Fourier transform), Filters(FIR-Finite Impulse Response),Signal Convolution and various other communication ,multimedia applications. The multiplier is the base of any arithmetic circuits. The multiplier consists of adders and shifters.

Multipliers were introduced to perform the multiplication operation of the arithmetic circuits using add and shift operation. A system's performance is generally determined by the performance of the multiplier because the multiplier is generally the slowest element in the whole system and also it is occupying more area consuming . Earlier multiplication was implemented via sequence of addition followed by subtraction and then shift operations [1]. An area-efficient parallel sign-magnitude multiplier that receives two N-bit numbers and produces an N-bit product, referred to as a truncated multiplier has been introduced[2].After some research it has been observed that the operations which share the data can be combined in the arithmetic circuits and performance can be increased[3]. The Multiply-Accumulator (MAC) and Multiply- Add (MAD) units were introduced as addition subsequent multiplication increasing the

DSP processor's efficiency [4][5]. A novel reconfigurable low-power, highperformance matrix multiplier design has been presented showing a large reduction in power dissipation compared [6]. Any multiplier can be divided into three stages: Partial products generation stage these are generated by AND operation, partial products addition stage can be carried by different adders, and the final addition stage. An area efficient Wallace tree multiplier is designed using common Boolean logic based square root carry select adder [7].Many DSP applications are based on Add- Multiply operations which was designed by adding the bits and giving its output as an input to the multiplier. This increases the area and delay of the circuit[8]. In order to reduce the power consumption of multiplier, the low power Booth recoding methodology is implemented by recoding technique.

III. PROJECT DESCRIPTION

The cascading nature of the ripple carry adder (RCA) slows down the carry chain, which increases the CPD of the multipliers. Bi-level concurrency followed in this work: (1) Booth radix-4 for PPG and (2) Dadda tree and a CSA for partial product addition (PPA). In the case of booth encoding (BE) based multipliers, the correct sign of the PPs will be determined by the multiplicand sign and corresponding BE value. Table I illustrates all possible Booth encoding values and combinations of MSB multiplicand sign extensions (SEs). We have employed Bewick's SE technique [6] to reduce the height of PP terms and optimized the sign extension. To avoid a compressor- based long carry chain, this work proposed an architecture with concurrent partial product generation

with Booth radix-4 and reduction with Dadda tree and CSA. Furthermore, compressors will consume a large area in multiplier design. Concurrent

PP reduction techniques like Wallace [14] and Dadda are efficient with CSA. Compared to the Wallace tree, the Dadda tree uses less number of half-adders [15]. Hence, it will reduce the CPD and power reduction effectively. This work concentrates on optimizing the logic slice's 6-input LUTs configurations for modern FPGAs (such as Xilinx Zynq and Virtex-7 series).

Signed Partial Product Generation

Configuration of the proposed multiplier with 6-input LUTs is depicted in Fig. 1(a). Standard Booth encoder implemented using the Type-A configuration. To produce partial product (Pout) 6-input LUT acquires five inputs i.e., multiplicand bits an, an−1 and multiplier bits bm+1, bm, bm−1. According to Booth encoder (BE) logic, as shown in Table I (a), selection line inputs (s, c, and z) generate PP terms through the selection lines of MUX. The first MUX (controlled by the s signal) determines output by selecting an or an−1 for PP generation. The first MUX's output is inverted by the second MUX is under the control of the c signal. The third MUX is controlled by the status of the z signal to make the PP zero. Finally, the Type-A block generates the partial product Pout.

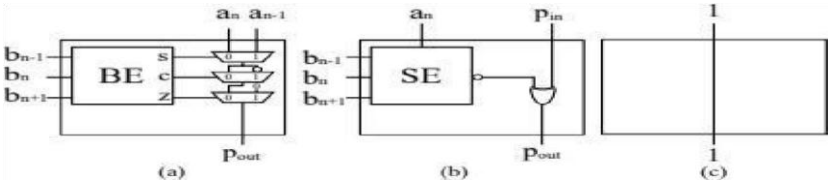


Fig. LUT configurations (a) Type-A. (b) Type-B. (c) Type-C [7].

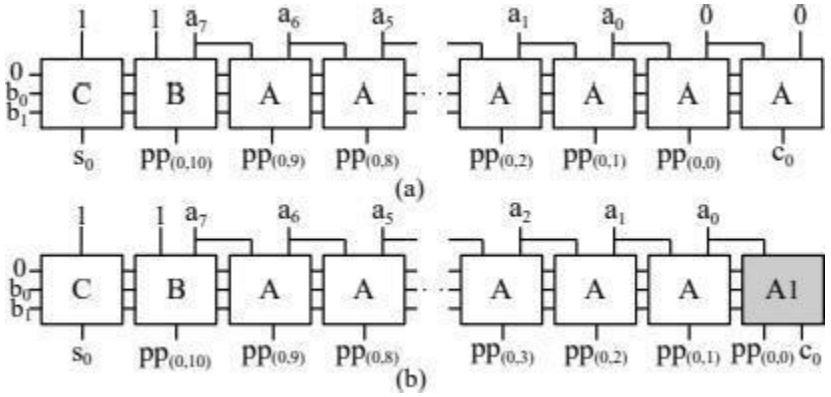


Fig. 8×8 multiplier's partial product for (a) Standard multiplier [7] (b) proposed multiplier.

LUT configurations of Type-B and Type-C are shown in Figure 1(b) and Figure
1(c), according to Bewick's signed extension (SE) approach for each PP row. Type-B configuration of LUT receives five inputs, i.e., multiplier bits bm+1, bm, and bm−1, the MSB of the multiplicand an, and Pin. For the first row of the PP, the Pin is constant '1', and for other rows, it is '0'. The SE signal is computed with LUT, and XOR is performed on it to generate associated the result Pout.

The arrangement of Type A, B, and C LUT configurations is shown in Figure 2 for the standard (except the first and last) row of PP for an 8×8 multiplier. The Type-A LUT at the rightmost is used to calculate the correction carry Cx and first PP bit PP(x,0).

This generated correction carry Cx is applied to represent the PP in 2's complement format. The value of Cx is '0' if PP is positive, otherwise '1'. PP signed bit, and its complement is shown with S and S, respectively, in Figure 4. The value of S is '1' if the multiplicand is positive and the partial product is positive, or if the multiplicand is negative and the partial product is negative, otherwise '0' according to Bewick's SE [6].

Stage 1 of Figure 4 shows Bewick's SEbased PP terms for an 8×8 multiplier.

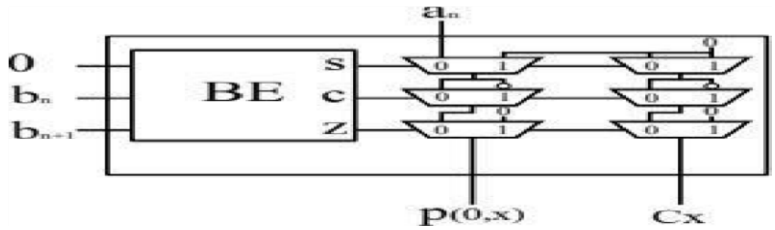


Fig. Proposed LUT configurations of Type-A1

Optimizing Critical Path Delay

For a multiplier with N×M bit-width, the carry chain length is calculated in (N+4) bits for each PP row [7]. To optimize the CPD, we carried out two strategies: (1) rightmost two LUTs shown in Figure 2(a) are combined and a block using 6-input 2- output LUT is generated, as shown in Figure 2(b). (2) Cx is handled by a PP reduction unit only, not with a long carry chain path. Hence, in this implementation, we do not compute the 2's complement, but we have generated the 1's complemented number with PPG and a Cx. Dadda algorithmbased adder tree and a CSA-based PP reduction unit handle the computation concurrently. Type A1 LUT takes a0 and '0' as the input and produces the output PP(0,0) and Cx as shown in Figure 3. The number of 6-input LUTs required to produce PPs for a N×M multiplier is (N + 3) × [M/2] − 1 [7].

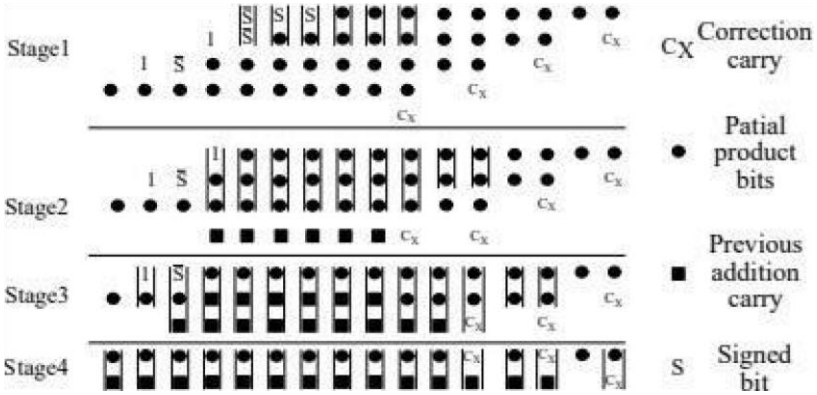


Fig. 8×8 Dadda-based partial product addition (PPA) [15].

Accumulation of Generated Partial Product

For multipliers, an array-like structure reduces the resource utilization at a cost of CPD [4]. A concurrent PPA-based Dadda tree adder is used in this work, which is faster than the Wallace tree [15]. If the time for the carry propagation is considered δc, then the time taken to perform multiplication is calculated by [(2N-2-logN)/log1.5] δc. All the generated PPs will be summed up here concurrently with CSA, which optimizes the CPD further. Half and full adders are implemented with 6-input 2-output LUTs and CARRY4 block of modern FPGAs. Booth radix-4 generated PPs are added with a Daddabased CSA and CPA for 8×8 inputs, as shown in Figure.

IV. SIMULATION RESULTS

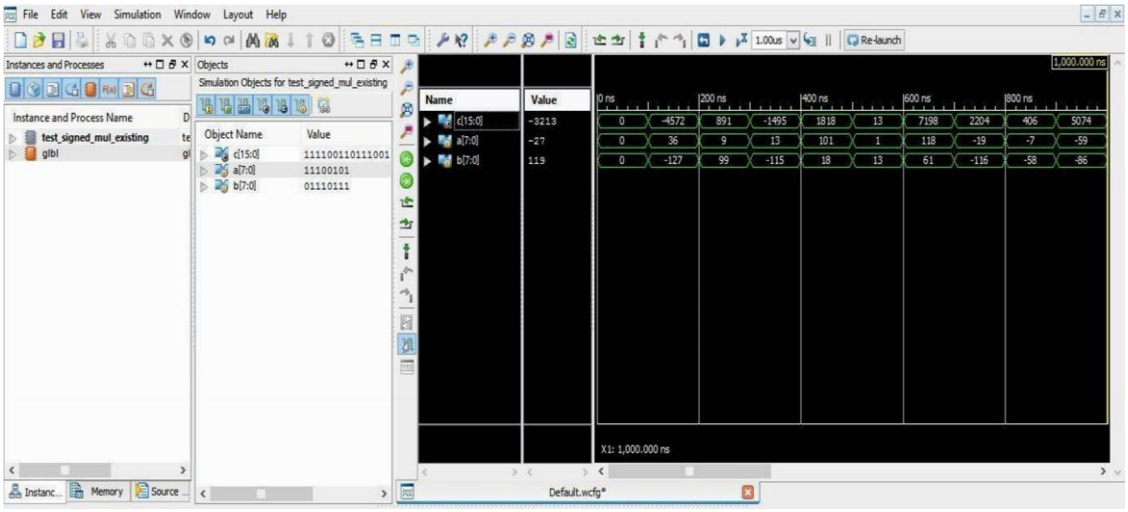


Figure: Simulation result of the Existing signed multiplier

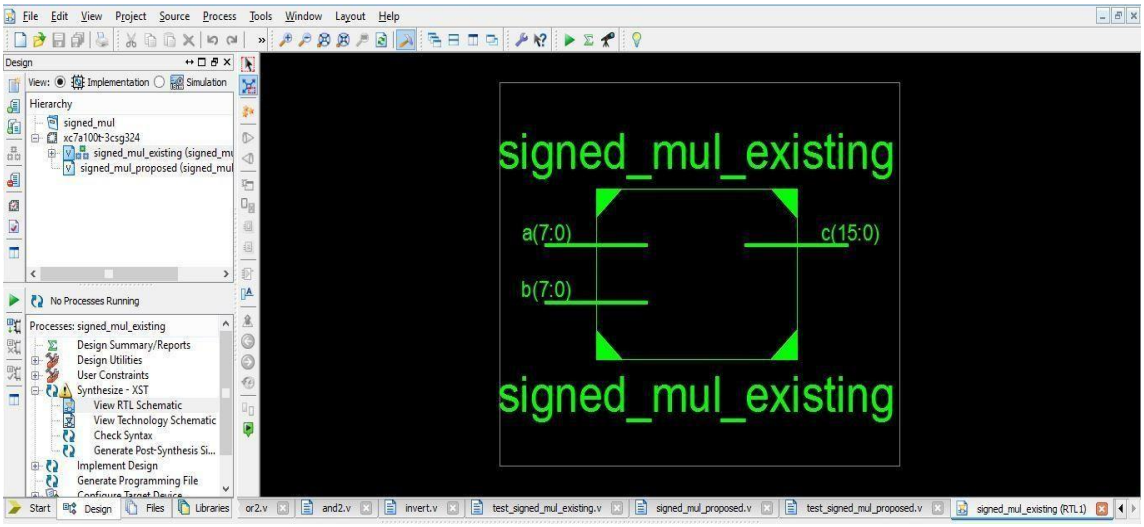


Figure: RTL Schematic of the Existing signed multiplier

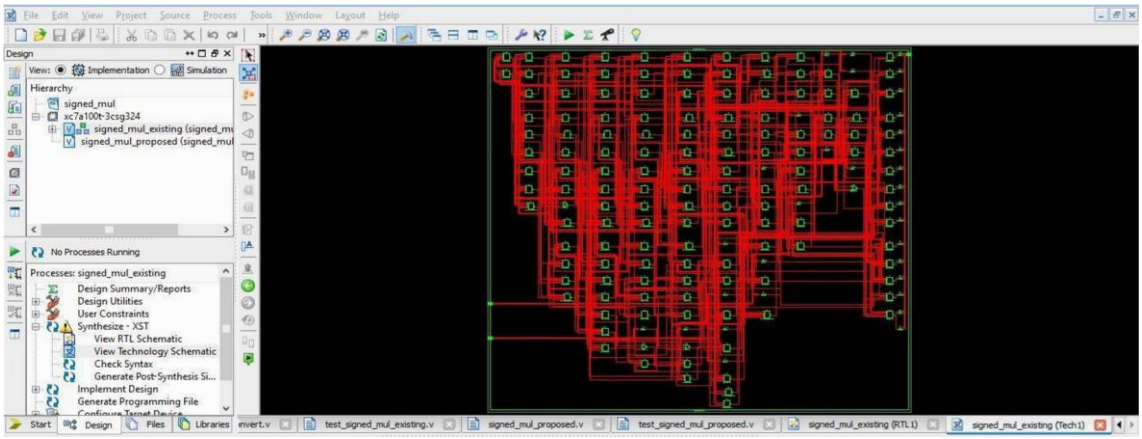


Figure: Technology Schematic of the Existing signed multiplier

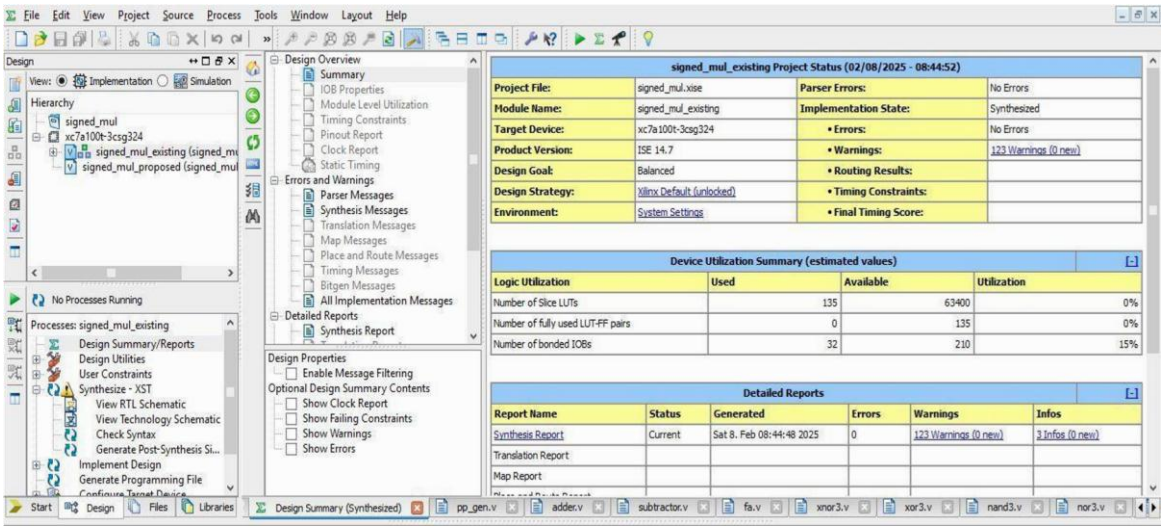


Figure: Summary report of the Existing signed multiplier

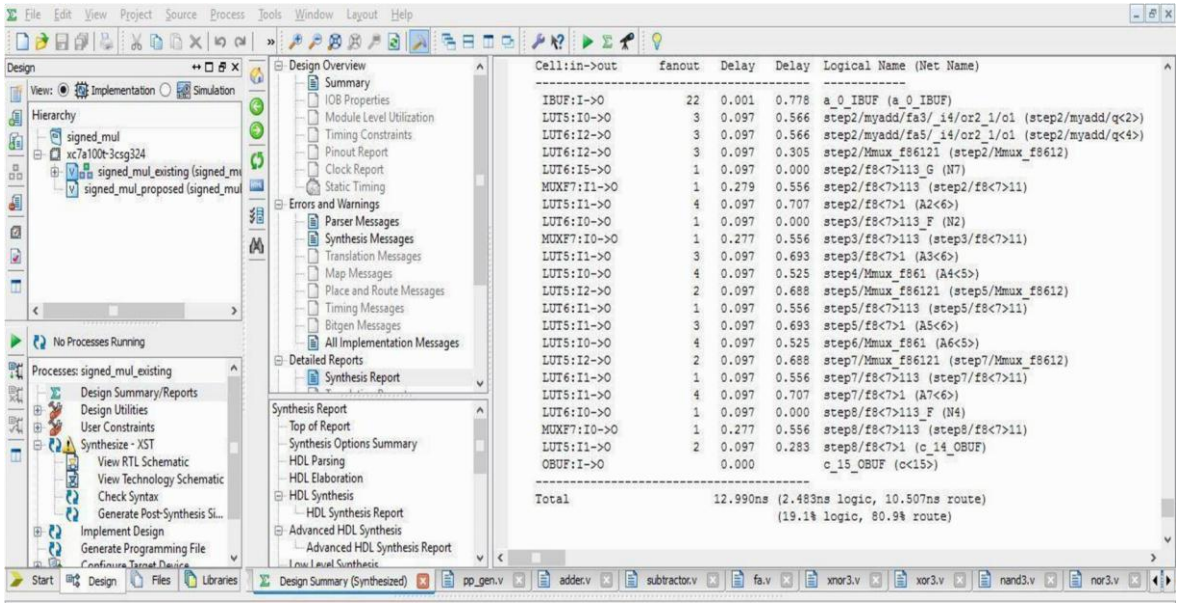


Figure: Synthesis Report of the Existing signed multiplier

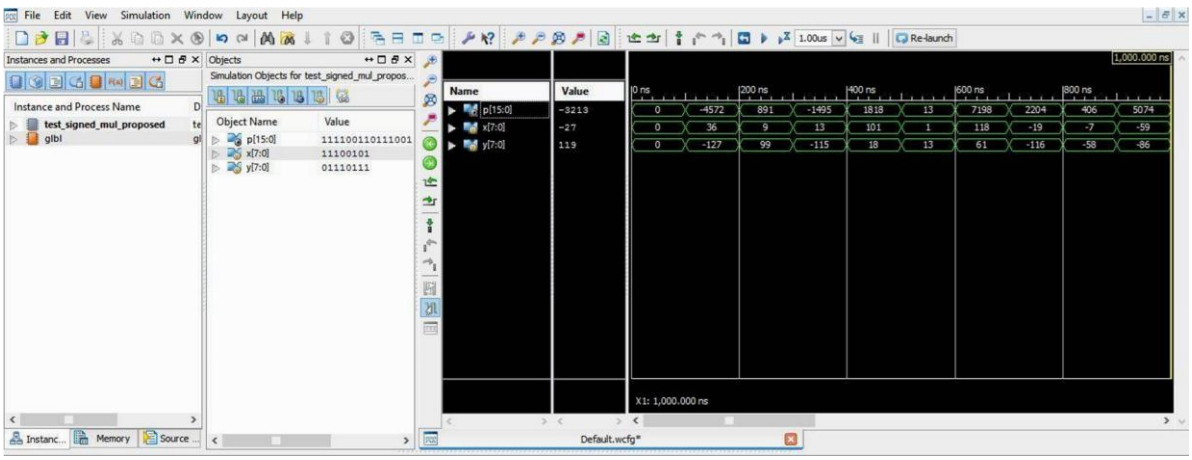


Figure. Simulation result of the Proposed signed multiplier

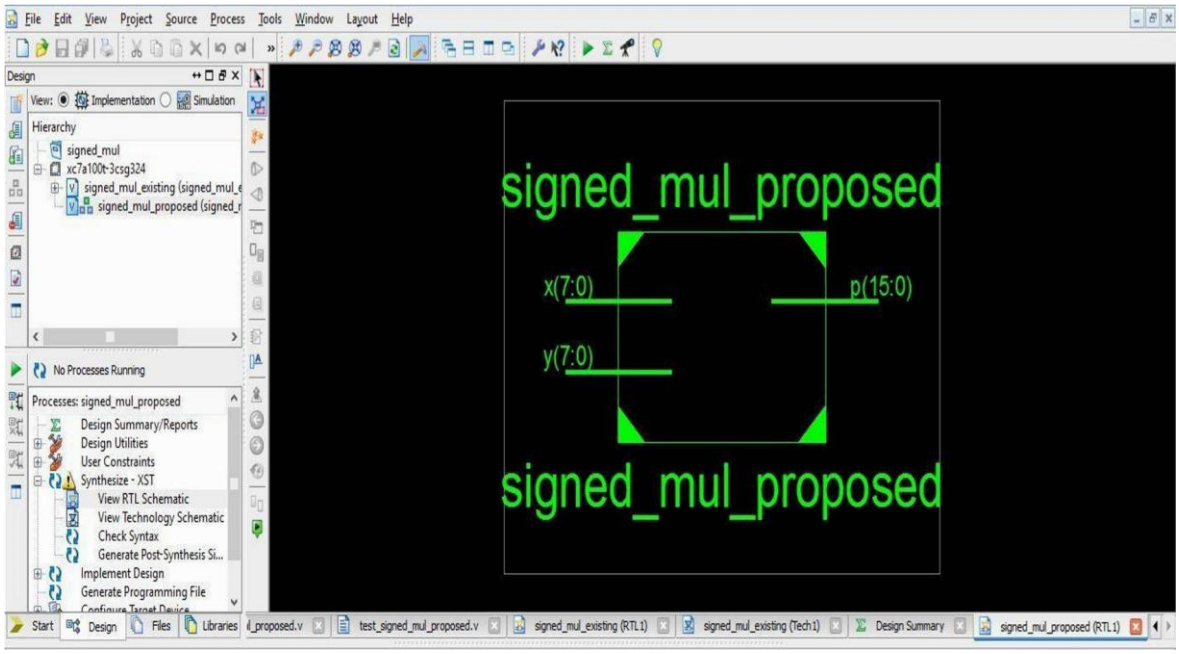


Figure. RTL Schematic of the Proposed signed multiplier

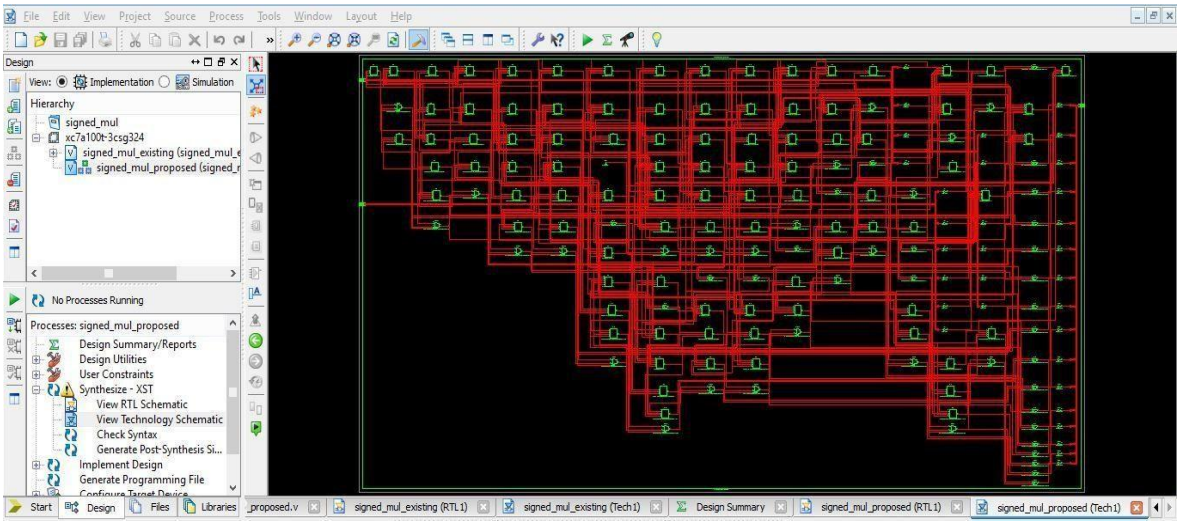
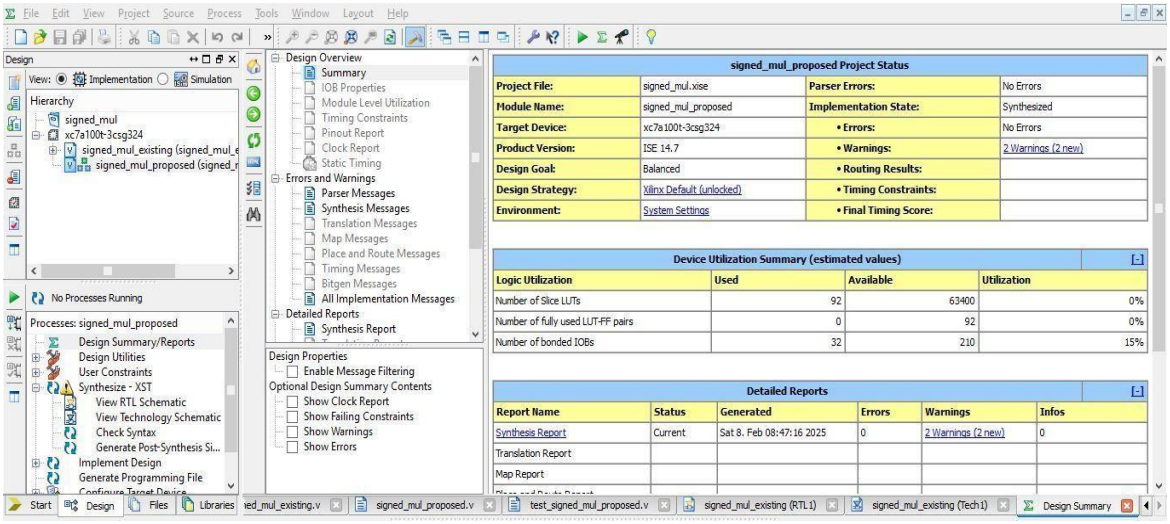


Figure. Technology Schematic of the Proposed signed multiplier



signed_mul_proposed Project Status			
Project File:	signed_mul.xise	Parser Errors:	No Errors
Module Name:	signed_mul_proposed	Implementation State:	Synthesized
Target Device:	xc7a100t-3csg324	• Errors:	No Errors
Product Version:	ISE 14.7	• Warnings:	2 Warnings (2 new)
Design Goal:	Balanced	• Routing Results:	
Design Strategy:	Xilinx Default (unlocked)	• Timing Constraints:	
Environment:	System Settings	• Final Timing Score:	

Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slice LUTs	92	63400	0%
Number of fully used LUT-FF pairs	0	92	0%
Number of bonded IOBs	32	210	15%

Detailed Reports					[-]
Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	Sat 8. Feb 08:47:16 2025	0	2 Warnings (2 new)	0
Translation Report					
Map Report					

Figure. Summary Report of the Proposed signed multiplier

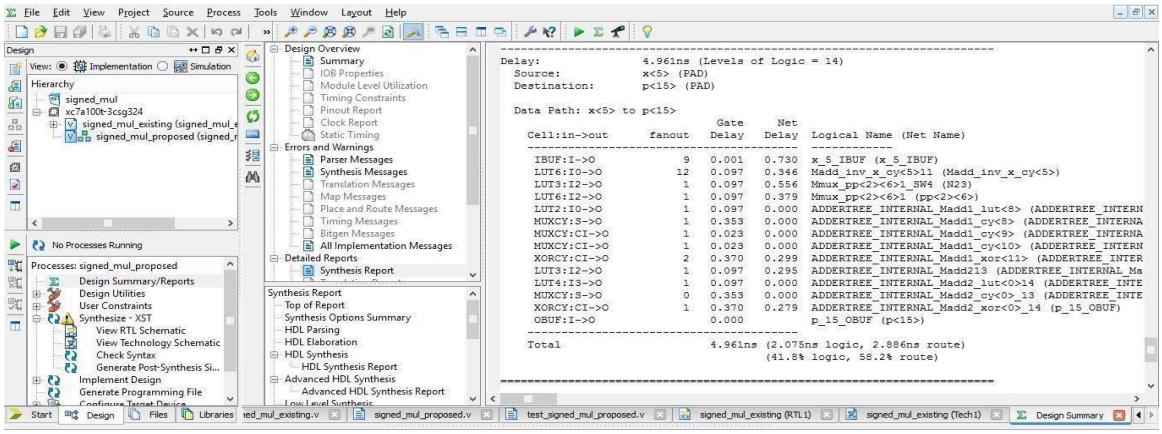


Figure. Synthesis Report of the Proposed signed multiplier

V. CONCLUSION

This paper proposes a novel high-speed and energy-efficient FPGA-based signed multiplier architecture. In this work, we have minimized the energy consumption by reducing the CPD of the multiplier. As per the analysis, the proposed 8×8 multiplier delivers the least energy consumption with minimum delay. By employing the proposed design (8×8 multiplier) for the convolution accelerator data path, it reduces the delay compared with the existing signed multipliers.

REFERRENCES

[1] A. Amaricai, M. Vladutiu, and O. Boncalo, "Design issues and implementations for floating- point divide-add fused," IEEE Trans. Circuits Syst. II–Exp. Briefs, vol. 57, no. 4, pp. 295–299, Apr. 2010.

[2] S. Ullah et al., "Area-optimized low-latency approximate multipliers for FPGA- based hardware accelerators", in Proc. ACM/ESDA/IEEE Design Autom. Conf. (DAC), 2018, pp.1-6.

[3] Intel. 2017, "Integer Arithmetic IP Cores User Guide. (2017)".

[4] LogiCORE IP Multiplier v12.0, document PG108, Xilinx, San Jose, CA, USA, 2015.

[5] M. Kumm, S. Abbas and P.Zipf, "An Efficient Software Multiplier Architecture for Xilinx FPGAs", IEEE 22nd Sympo. on Computer Arithmetic, Lyon, France, 2015, pp. 18- 25.

[6] E. E. Swartzlander and H. H. M. Saleh, "FFT implementation with fused floating- point operations," IEEE Trans. Comput., vol. 61, no. 2, pp. 284–288, Feb. 2012.

[7] J. J. F. Cavanagh, Digital Computer Arithmetic. New York: McGrawHill, 1984.

[8] S. Nikolaidis, E. Karaolis, and E. D. Kyriakis-Bitzaros, "Estimation of signal transition activity in FIR filters implemented by a MAC architecture," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 19, no. 1, pp. 164–169, Jan. 2000.

[9] O. Kwon, K. Nowka, and E. E. Swartzlander, "A 16-bit by 16-bit MAC design using fast 5: 3 compressor cells," J. VLSI Signal Process. Syst., vol. 31, no. 2, pp. 77–89, Jun. 2002.

[10] L.-H. Chen, O. T.-C. Chen, T.-Y. Wang, and Y.-C. Ma, "A multiplication- accumulation computation unit with optimized compressors and minimized switching activities," in Proc. IEEE Int, Symp. Circuits and Syst., Kobe, Japan, 2005, vol. 6, pp. 6118–6121.