ISSN: 2321-2152

IJMECE

International Journal of modern electronics and communication engineering

E-Mail
editor.ijmece@gmail.com
editor@ijmece.com

www.ijmece.com

ISSN2321-2152

www.ijmece .com

Vol 12, Issue 2, 2024

High Speed VLSI Implementation FIR Filter of Odd Length: Leveraging BK Adder and Booth Multiplier

Mrs. T Ashwini (Asst.prof), Priya Nalla , Sripati Manoj Kumar, and Samanuri SriHari Varma
Department of ECE
Sreyas Institute of Engineering and Technology
9-39, Sy No107 Tattiannaram, beside INDU ARANYA HARITHA,
Bandlaguda , Nagole , Hyderabad, Telangana500068, India
nallapriya28@gmail.com

***Abstract*—The fundamental principles guiding the planning and execution of DSP processors revolve around the reduction of power consumption and optimizing chip area. At the heart of this strategy lies the design, implementation of a Parallel FIR filter, a critical component in DSP systems. This paper introduces the utilization of a Feed Forward Architecture (FFA) based 3-equal poly-phase FIR filter, incorporating adder and multiplier structures in place of conventional ones.**

We introduced a three-parallel poly-phase FIR filter is proposed in this study, utilizing two different types of multipliers: the Vedic multiplier and the Booth multiplier. Additionally, three distinct adders, namely the Ripple Carry Adder, Carry Look-ahead Adder, and Brent Kung Adder, are integrated. This incorporation of advanced components is aimed at improving both the efficiency and performance of the FIR filter.

The FIR filter implements using various multipliers and adders, and a comprehensive comparison is conducted based on different parameters. This evaluation encompasses considerations for both delay and area. Notably, the Booth multiplier and Brent Kung adder demonstrate significant advantages, showcasing reduced power consumption and delay, making them highly useful for the design of parallel FIR filters intended for low-power and compact VLSI applications. This innovative approach aligns with the overarching goal of achieving optimal performance while minimizing power utilization and chip area in DSP processors.

Keywords—Brent Kung adder, Booth Multiplier, Ripple carry adder, Vedic multiplier, FIR Filter, etc.,

I. INTRODUCTION

DSP is a critical element in numerous applications such as communications, audio processing, and image processing. Finite Impulse Response (FIR) filters serve as fundamental components in DSP systems, tackling tasks like signal filtering and noise reduction. The efficiency of FIR filters is paramount for real-time processing, driving ongoing research into innovative architectures to enhance performance. This project introduces a High-Speed VLSI Implementation of an Odd-Length FIR Filter, capitalizing on advanced arithmetic units for optimized speed and area utilization.

The design incorporates a BK adder and a Booth multiplier, both renowned for their efficiency in high-speed arithmetic operations. The BrentKung adder, a parallel prefix adder with a regular structure and efficient carry propagation mechanism, facilitates rapid addition of multiple operands simultaneously, contributing to the overall speed enhancement of the FIR filter. Meanwhile, the Booth multiplier is selected to reduce the partial products generated during multiplication, resulting in a more streamlined and faster multiplication operation. This is particularly advantageous in FIR filters, where a substantial portion of computation involves multiplication.

The project entails a thorough exploration of the architectural design, Verilog coding, and subsequent synthesis and optimization steps. The proposed FIR filter is implemented on a VLSI platform, emphasizing the use of modern technologies for efficient resource utilization and high-speed processing.

The methodology encompasses the selection of appropriate filter specifications, the design of individual components, and the seamless integration of the BK adder and Booth multiplier into the FIR filter structure. The performance of the design undergoes rigorous evaluation through simulations and post-layout analyses. This research significantly contributes to the field of VLSI-based FIR filter design by synergizing the strengths of a high-speed adder and multiplier. The resulting benefits include reduced computation time, rendering it suitable for applications demanding rapid and efficient signal processing. Subsequent sections of this document will delve into the detailed design, implementation, and performance analysis of the proposed High-Speed FIR filter

II. EXISTING SYSTEM

The half adder is a fundamental binary addition component that takes two binary digits, A and B, producing two outputs: sum (S) and carry (C). The sum is given by the Boolean expression A'B + AB', while the carry is determined by AB. By combining two half adders with an OR gate, a full adder is created, allowing for the addition of 3 1-bit numbers – A, B, and a carry-in bit (Cin). The full adder, typically part of a cascade of adders for processing multi-bit binary numbers, gives a two-bit output represented by Cout and S, where the sum equals 2Cout + S.

ISSN2321-2152

www.ijmece .com

Vol 12, Issue 2, 2024

Alternatively, full adder can be made by two half adders by joining A and B to the input of the second half adder, using its sum output(s) with Cin as one input of the second half adder. The carry outputs of both sides of the adder are then connected to an Or gate, where the sum of the adder on the adder on the other side is the full adder and final sum output(s) and the output of the OR gate being the final carry output(Cout). The critical path of the full adder contains XOR gates and the delay depends on the XOR gate and the AND and the OR gates of the transfer block.

For an RCA consisting of n consecutive full adders of an n-bit adder, the corresponding logic equations are gi=ai biand pi=ai ⊕bi. The delay(T)-wavelet transfer viewer and area(A)complexity is given by ARCA=O(n)=7n and TRCA=O(n)=2n.

In a port-selected sump, blocks of bits are added in two ways, assuming the carry is 0 and 1. Precomputed sum and port signals pairs are generated, and if the actual port(ck) of the block is known, corrected signal pairs is selected. The cargo-looking technique, on other hand aims to speed up transmission in the summer by generating all incoming transmissions in parallel through additional logic circuits indicated by thr propagate(pi) and generate (gi) signals.

Several other multibit adder architectures divide the adder into blocks, optimizing computation time based on propagation delay. Examples include Carry-skip(carry-bypass), which determines the P and G values for each block, and carry-store, which calculates the sum of three or more numbers regardless of the result of the carry, using a conventional view for to combine final sum and transfer results after all stages of aggregation.

III. PROPOSED SYSTEM

The project focuses on enhancing the speed and efficiency of Finite Impulse Response (FIR) filters, crucial components in DSP. The specific innovation lies in leveraging a Brent Kung adder and a Booth multiplier for high-speed arithmetic operations.

A. Booth Multiplier

The selection of the Booth multiplier for this project is rooted in its adept ability to optimize the multiplication process. The Booth encoding technique, a distinguishing feature of this multiplier, plays a crucial role in enhancing efficiency and accelerating multiplication operations. This technique strategically reduces the partial products generated during the multiplication process, resulting in a more streamlined and expedited overall operation. Considering that multiplication is a fundamental operation in FIR filters, the efficiency gained through the Booth multiplier holds substantial importance, directly contributing to the improvement of the system.

B. Brent Kung Adder

This adds a parallel prefix known for its regular structure and efficient carry propagation mechanism. It allows for simultaneous addition of multiple operands, reducing overall computation time. Regularity in its design facilitates easier integration into complex systems like FIR filters.

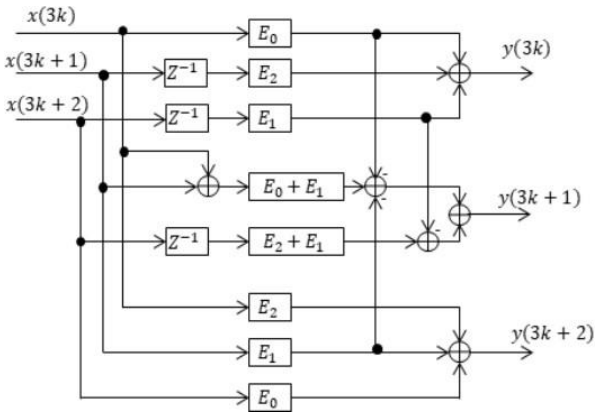

Fig 1: Odd-Length FIR Filter

The embedding process includes the following three stages:

a)Pre-processing step.

b)Carry generation network.

c)Post-processing step.

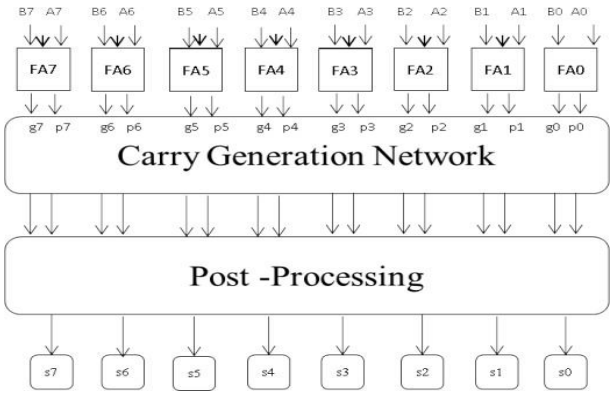

Fig 2: Bk adder steps

IV. FLOW OF VLSI DESIGN

The VLSI circuit design flow, as illustrated in tool flow diagram, serves as a systematic process widely adopted by designers employing Hardware Description Languages (HDLs). The journey begins with the definition phase, where digital scheme functionality, user interface and overall architecture are well defined of. Architects focus solely on conceptualizing the design without delving into implementation details. Subsequently, behavioral description is prepared to comprehensively analyze the design functionality, performance, and high-level considerations. This behavioral description is manually translated into a Register Transfer Level (RTL) description in an HDL, wherein the designer outlines the data flow to realize the desired digital circuit.

ISSN2321-2152

www.ijmece .com

Vol 12, Issue 2, 2024

At this juncture, Computer-Aided Design (CAD) tools come into play, facilitating the logic synthesis process. Logic synthesis tools convert the RTL(Register Transfer Level) description to a port-level net-list, presenting the circuit's representation through interconnected gates. The gate-level net-list serves as input for an automatic place and route tool, generating a layout for the circuit. Rigorous verification of the layout follows, leading to the fabrication of the circuit on a chip.

The crux of digital design lies in the manual optimization of the RTL description, a phase that significantly influences the overall design cycle. Once the RTL description is finalized, CAD tools continue to assist in subsequent processes. Designing at the RTL level has revolutionized design cycle times, reducing them from years to just a few months.

Verilog HDL emerges as a powerful tool in this design flow. As a Hardware Description Language, Verilog supports various levels of abstraction, from specifying the layout of wires, resistors, and transistors (switch level) to describing logical gates and flip-flops (gate level) and even higher levels like Register Transfer Level (RTL). Verilog's versatility allows designers to use the same language for describing, testing, and debugging systems, providing a cohesive and efficient design environment.

V. XILINX ISE 14.7 AND DESIGN SIMULATION

To run a new ISE project targeting an FPGA device with demo board Spartan-3 Startup Kit , do the following:

1. Open the new project wizard by selecting File>new project.

2. Type “tutorial” in the project Name Field.

3. Specify the location of the bew project or browse to the folder path…The tutorials subfolder will be created automatically.

4. Select HDL from top-level source type list.

5. Click Next to go to the device properties page.

6. Complete the properties table as follows:

Product Category: All

(family)Spartan3

(device)XC3S200

(package) FT256

(speed) -4

HDL

(synthesis tool) XST (VHDL/Verilog)

(simulator) ISE Simulator (VHDL/Verilog)

(language) Verilog (or VHDL)

select the "Enable Enhanced Design Summary" option.

7. default values in the leftover fields.

Using language templates (Verilog)

1. Start by placing the cursor under the output [3:0] COUNT_OUT; expression in the source file.

2. Access the Language Templates by navigating to Edit → Language Templates…

3. For better visibility, The language Templates and counter file can be assembled choosing Window → Pane vertically.

4. Employ the "+" symbol to navigate through Verilog ->Synthesis Constructs -> Coding Examples -> Counters -> Binary -> Up/Down Counters, and examine the provided code example for a Simple Counter.

5. Once you have identified a simple number example, choose Edit -> Use in File, or use the Use template File button on the toolbar. This function effectively imports the template into the next source file.

6. Close language templates window.

Following these steps ensures the seamless addition of a behavioral description for the counter in the source file, utilizing a customized and appropriate code example from the ISE Language Templates.

Design Simulation

1. Start by selecting the anti-HDL file in Sources window.

2. Write new test bench source by showing to Project -> new source.

3. In new source wizard, set the source type to “test bench waveform” and name it “counter” in file name field.

4. Advance to subsequent stage by selecting the "Next" option.

5. On the Related source page, confirm you are adding the testbench waveform to the counter source file, and then click Next.

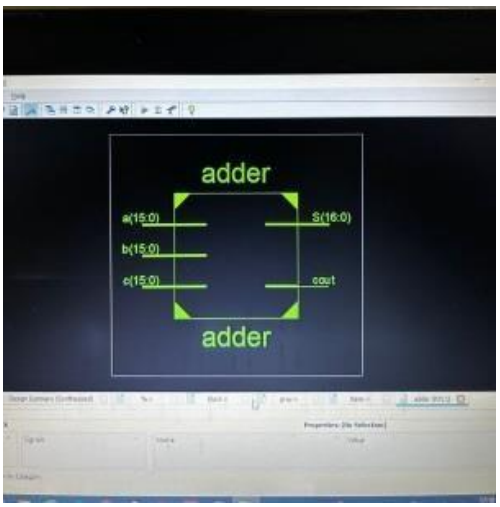

Fig 3: Adder block

ISSN2321-2152

www.ijmece .com

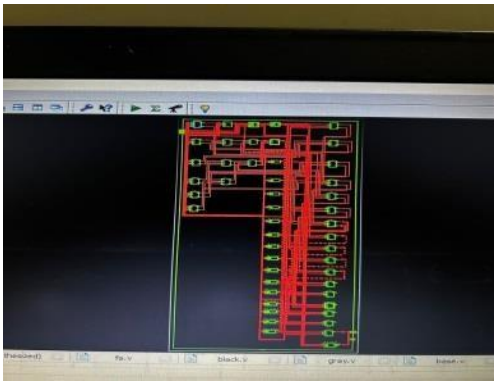

Fig 4: RTL schematic

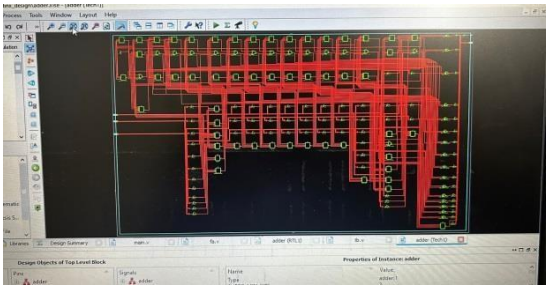

Fig 5: Technology schematic

After writing the testbench , that is after implementation we have to proceed for simulation. We get the following result for bk adder.

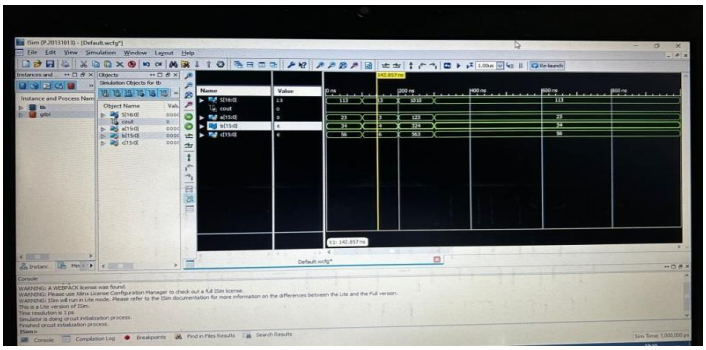

Fig 6: Simulation of bk adder

Same goes with the booth multiplier.

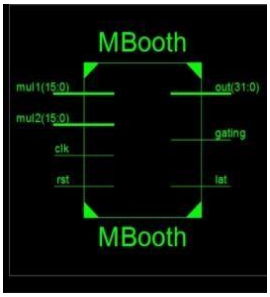

Fig 7: multiplier block

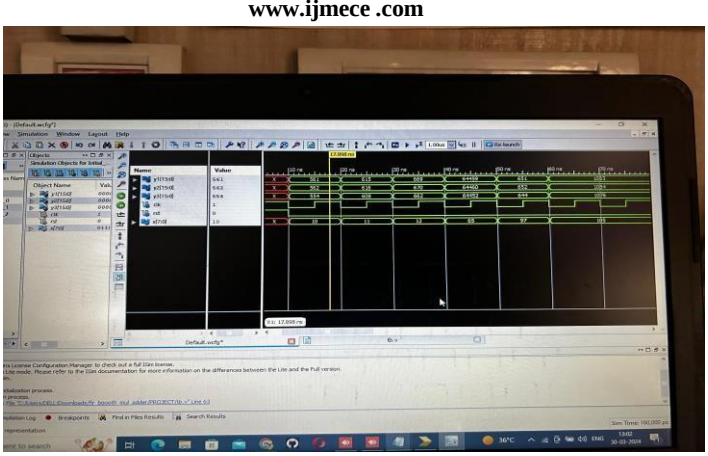

Fig 8: simulation of multiplier and final result of fir..

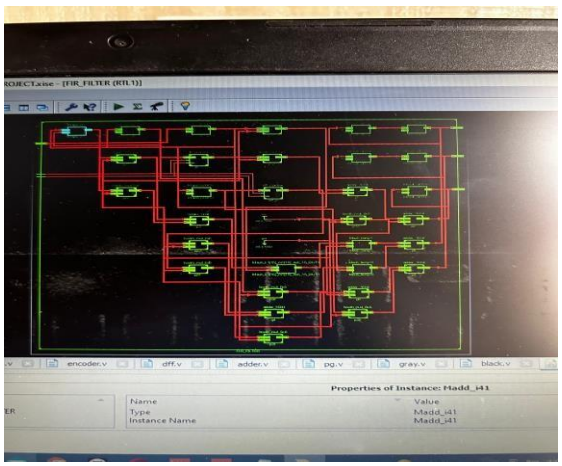

Fig 9: RTL schematic

6. The last(summary) page will displays details of the new source to be added to the project. Confirm and click Finish.

7. Before entering the test bench waveform editing window, set the clock frequency, setup time and output delays in the Time Reset dialog according to the design requirements.

(Clock frequency) 25 MHz.

The direction input is valid for 10ns before the rising edge of the clock.

Count_out output valid for10ns after the rising edge of clock.

8. Complete the timing reset dialog with the given information:

(clock high time) 20 ns.

(clock low time) 20 ns.

(input setup time) 10 ns.

(output valid delay) 10 ns.

(offset) 0 ns.

ISSN2321-2152

www.ijmece .com

Vol 12, Issue 2, 2024

9. Set the initial length of testbench -> 1500 ns. Conclude the timing initialization by clicking the "Finish" button.

10. Observe the blue areas in front of rising edge of clock, which represent input configuration time specified in Reset Time dialog.

11. Specifying input to draw a counter, toggle the DIRECTION port as follow:

select the blue cell approx 300 ns to set direction high, instructing the Counter to count up.

select the blue cell approx 900 ns to set direction low, prompting the Counter to count down.

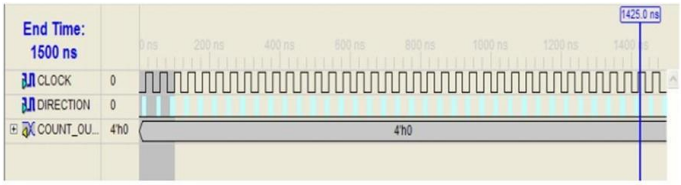

Fig 10: clock pulses

Generally our project main applications are for audio and video processing, telecommunications, Biomedical Engineering and speech processing, recognition.

VI. CONCLUSION

In conclusion, the project aimed at achieving a high-speed VLSI implementation of an FIR of odd length filter has been successfully completed. Leveraging the strengths of the BK adder and booth multiplier, we addressed the critical need for efficient and speedy FIR filtering in dsp applications. Use of Brent Kung adder contributed significantly to the speed enhancement of the FIR filter. The parallelism in its structure facilitated faster addition, crucial for real-time signal processing. The design's efficient carry look-ahead mechanism minimized delays associated with carry propagation, a key factor in achieving high-speed computation. The inclusion of the Booth multiplier complemented the Brent Kung adder by optimizing the multiplication stage of the FIR filter. The Booth multiplier's ability to reduce the number of partial product additions positively impacted overall filter performance.

ACKNOWLEDGMENT

We would like to thank Sreyas Institute of engineering and technology, Hyderabad for readily available research facilities and computer resources.

REFERENCES

[1] K. K. Parhi,VLSI Digital Signal Processing System : Design and Implementation (Wiley, New York, 1999) .
[2] L. K. Phimu and M. Kumar, “Design and implementation of area efficient 2-parallel filters on FPGA using image system” in proc. ICECDS, 2017.
[3] Z.-J. Mou, and P. Duhamel, “Short-length FIR filters and their use in fast nonrecursive filtering”IEEE Trans. Signal Process., vol. 39, no. 6, 1991.
[4] D. A. Parker, and K. K. Parhi, “Low-area/power parallel FIR digital filter implementation,” J. VLSI Signal Process. Syst, vol. 17, 1997..
[5] Q. Tian, Y. Wang, G. Liu, X. Liu, J. Diao, and Hui Xu “Hardware-efficient parallel FIR filter structure based on modified Cook-Toom algorithm ”in proc. IEEE APCCAS, 2018.
[6] K Anjali Rao, Abhishek Kumar, Neetesh, “Efficient implementation for 3-parallel linear- phase FIR digital odd length filters ” in proc. IEEE CICT, 2020 .
[7] A.Kumar, S. Yadav and N. Purohit, “Exploiting coefficient symmetry in conventional polyphase FIR filters,” IEEE Access, vol.7, 2019.
[8] S.Y. Park, and Pramod K. Meher, “Efficient FPGA and ASIC realizations of DA-based reconfigurable FIR digital filter,” IEEE Trans. Circuit and Syst.II, vol.61, no. 7, 2014.
[9] T. Vamshi Krishna, Niveditha S, Mamatha G. N, Sunil M. P. “Simulation study of brent Kung adder using cadence tool,” International Journal of Advanced Research, Ideas, and Innovations in Technology, 2018.
[10] D. K. Kahar and H. Mehta, “High-speed Vedic multiplier used in Vedic mathematics, ” in Proc. ICICCS, 2018.
[11] Rajesh K., Reddy G. “FPGA based implementation of multiplier accumulator unit by using Vedic multiplier and reversible gates ” in proc. ICISC, 2019.
[12] S. Nagaria, A. Singh, and V. Niranjan “Efficient FIR filter design using Booth multiplier for VLSI applications ” in proc. IEEE CPCT (GUCON), 2018 Authorized.
Combine duplicate citations for the same source to enhance clarity and conciseness:

[13] J. Selvakumar and Vidhyacharan, "Efficient complexity reduction technique for parallel FIR digital Filter based on Fast FIR algorithm," International Journal of Computer Applications, Vol. 55, 2012.

[14] Y.C. Tsao and K. Choi, "Hardware-efficient VLSI implementation for 3-parallel linear-phase FIR digital filter of odd length" in Proc. IEEE ISCAS, 2012.