INTERNATIONAL JOURNAL OF TECHO-ENGINEERING

ISSN : 2057-5688

Design of Efficient VLSI Arithmetic Circuits

N.L.Sowmya[1] Dr. S.V.S PRASAD[2]

[1, 2] Department of ECE, *MLR INSTITUTE OF TECHNOLOGY*, Hyderabad, Telangana, India.

E-mail: [1] sonu.nadimpally@gmail.com, [2] prasad.sista@gmail.com, [3]hodece@mlrinstitutions.ac.in

ABSTRACT:

Arithmetic and Logic Unit (ALU) is the critical component of any CPU. Adders are important in ALU since they are used not only for addition but also for many other basic arithmetic operations such as multiplication, subtraction, and so on. As a result, developing an efficient adder is essential to improve the performance of an ALU and, as a result, the processor. The development of efficient adder algorithms and their hardware implementation began in the late 1950s. To maximise distinct parameters over time, many designs based on serial and parallel structures have been developed. The initial contribution of this thesis is the creation of an efficient adder architecture that handles issues such as fan-out, wiring complexity, and other issues that arise with longer bit operand lengths. After the adder, the multiplier was a significant component in an ALU. Multi-operand adders and fast adders are necessary in multipliers for lowering partial products and computing the final result. For scheming multi-operand adders, a particular structure known as counters/compressors is often employedCounters are multi-input, multi-output combinational logic circuits (CLCs) that calculate the OUTPUT of logic 1s in their input vectors and generate a binary coded output vector that corresponds to that number. Large parallel counters such as (15, 4), (32, 5), and others can be built using this little counter, and compressors can be built in the same way. The thesis' second contribution is the creation of efficient counters and compressors to improve multiplier performance. The thesis' second contribution is the creation of efficient counters and compressors for improved multiplier performance. Because of the rise in decimal data processing in commercial, financial, and internet-based applications, the requirement for hardware support for decimal arithmetic has grown in recent years. The design of a multi-functional INCREMENT/complement/Priority DECV/2's encoder circuit is the thesis' third contribution. A design for binary INCREMENT/DECREMENTs that is efficient in terms of speed without sacrificing power is shown. A reconfigurable technique is required to simplify binary computations on the same hardware. The invention of a new architecture for efficient Binary Coded Decimal (BCD) addition/subtraction that can also do binary addition/subtraction is the thesis' fourth contribution. The architecture was created with the signed magnitude format in mind, where the adder logic recognises the larger operand and performs the necessary operations.

Finally, unique versions of two often used arithmetic blocks, namely the multiplier and floating point adder, are created. These blocks are implemented using the above-mentioned efficient and proven basic functional components. Simulations of these blocks have been run, and comparisons with existing designs have been done, demonstrating that the suggested units are highly efficient. Finally, all of the foregoing features are integrated into a segment of a processor's core, resulting in an efficient architecture.

KEYWORDS: Adder, fan-out, wiring complexity, counters , compressors, floating point adder and Multiplier

I INTRODUCTION

Data path efficiency is critical in microprocessors and digital signal processors (DSPs), because performance measurements like device area, speed of operation, power dissipation, and others are all directly reliant on it. The data path's core, as is generally known, comprises of sophisticated operations including addition, subtraction, multiplication, and division. As a result, it's vital to create efficient hardware units for these computations, which have a direct impact on data path performance. The incrementer/decrementer (INC/DEC) block can conduct the increment and decrement operations, which count up or down by one stepThis block is also used in processors' address generating units and frequency dividers. Adder/subtractor, counter, or carry look-ahead adder are the most common architectures for binary INC/DEC blocks.

The main objectives are

- Efficient detection of bit adders with higher operands (for 32-bit and above).
- Development of efficient counters /compressors for parallel multiplication at high speeds.
- Development of high-performance standalone blocks, such as increment and decrement.
- The use of a unified Binary/BCD adder will result in improved performance.

- Use of the proposed basic units to demonstrate an ALU's efficient arithmetic section.

II LITERATURE SURVEY

R. Hashemian and C. P. Chen. "A New Parallel Technique for Design of Decrement/Increment and Two's Complement Circuits." in Proceedings of the 34th Midwest Symposium on Circuits and Systems, volume 2, pages 887-890, 1991. [1]

Shaoqiang Bi, Wei Wang, and Asim Al-Khalili, "Multiplexer-based Binary Incrementer/decrementers," The 3rd International IEEE-NEWCAS Conference,19-22 June 2005. pp. 219-222. [2]

Veeramachaneni, S. Avinash, L. Kirthi, K.M. Srinivas, M.B. "A Novel High-Speed Binary and Gray Incrementer/Decrementer for an Address Generation Unit", International Conference on Industrial and Information Systems (ICIIS), 9-11 August 2007. pp.427-430. [3]

H. Fischer and W. Rohsaint. "Circuit Arrangement for Adding or Subtracting Operands Coded in BCD-Code or Binary-Code", Siemens Aktiengesellschaft, US patent 5146423, pages 1 – 9, Sep 1992. [4]

D.R.Humberto Calderón, G. N. Gaydadjiev, S. Vassiliadis, "Reconfigurable Universal Adder", Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP 07), pages 186-191, July 2007. [5]

Chetan Kumar, V. Sai Phaneendra, P. Ahmed, Syed Ershad, Veeramachaneni, Sreehari; Muthukrishnan, N. Moorthy; Srinivas, M.B. , "A Unified Architecture for BCD and Binary Adder/Subtractor," Digital System Design (DSD), 2011 14th Euromicro Conference on , vol., no., pp.426-429, Aug. 31 2011-Sept. 2 2011. [6]

H. H. Saleh and E. E. Swartzlander, Jr., "A Floating-Point Fused Add-Subtract Unit," Proceedings of the 51st IEEE Midwest Symposium on Circuits and Systems, 2008, pp. 519 - 522. [7]

Jongwook Sohn, Earl E. Swartzlander Jr. "Improved Architectures for a Fused Floating-Point Add-Subtract Unit". IEEE Trans. on Circuits and Systems 59-I(10): 2285-2291 (2012) [8]

JongwookSohn "Improved architectures for a fused floating-point add-subtract unit", M.S. Thesis , The University of Texas at Austin,2011. [9]

H.H. Saleh, "Fused Floating-Point Arithmetic for DSP", PhD dissertation, Univ. of Texas, 2009. [10]

W.J. Townsend, E.E. SwartzlanderJr., and J.A. Abraham, "A Comparison of Dadda and Wallace Multiplier Delays," *Proc. SPIE,* Advanced Signal Processing Algorithms, Architectures, and Implementations XIII, pp. 552-560, 2003. [11]

Ron S. Waters, Earl E. Swartzlander, "A Reduced Complexity Wallace Multiplier Reduction", IEEE Transactions on Computers, vol. 59, no. 8, pp. 1134-1137, Aug. 2010. [12]

III Increment/Decrement circuits

In the literature, there are several designs for binary INC/DEC circuits[1-3]. The increment/decrement action is implemented in many of these systems using an adder. By making one input operand and the other input '1', an adder/subtractor can be utilised for these operations. However, when compared to a dedicated INC/DEC design, using the full adder for single bit addition adds time and power. As shown in Fig 1 (a) , the current designs of binary INC/DEC are primarily adder/subtractor-based, counter-based, or Carry look-ahead adder-based[1]. As illustrated in Fig 1 (b) , a MUX-based binary INC/DEC that is more efficient than earlier INC/DECs has recently been presented in the literature[2]. A data-in MUX array, a decision block, and a data-out MUX array are all included in this circuitWhile this design is efficient in terms of both performance and hardware complexity when compared to adder-based alternatives, the critical route has a series of (n-2) OR gates and a MUX, which slows the circuit down to some amount. In the

ISSN : 2057-5688

literature, there are several designs for binary INC/DEC circuits[1-3The increment/decrement action is implemented in many of these systems using an adder. By making one input operand and the other input ‘1', an adder/subtractor can be utilised for these operations. However, when compared to a dedicated INC/DEC design, using the full adder for single bit addition adds time and power. As illustrated in Fig 1 (a) , the present designs of binary INC/DEC are primarily adder/subtractor-based, counter-based, or Carry look-ahead adder-based[1]. As illustrated in Fig 1 (b) , a MUX-based binary INC/DEC that is more efficient than earlier INC/DECs has recently been presented in the literature[2]. A data-in MUX array, a decision block, and a data-out MUX array are all included in this circuit. While this design is efficient in terms of both performance and hardware complexity when compared to adder-based alternatives, the critical route has a series of (n-2) OR gates and a MUX, which slows the circuit down to some amount.When compared to MUX-based INC/DEC, an enhancement to this circuit was described in [3], which uses a look-ahead type technique and results in reduced delay. When an 8-bit look-ahead block is employed, an N-bit Carry look-ahead type results in (N/8) + 9 gates delay.

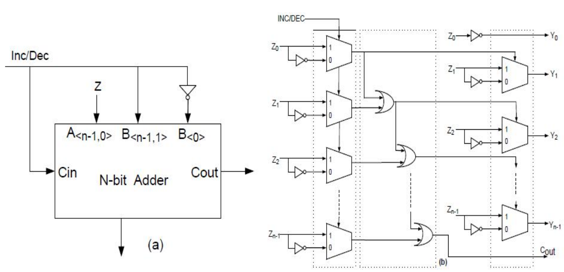

Figure 1 (a) Adder based (b) MUX based

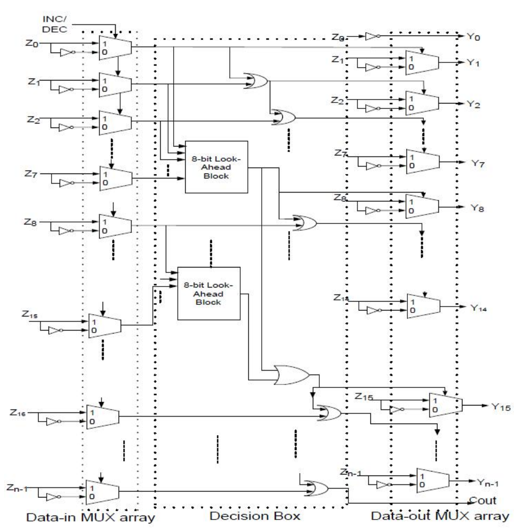

Figure 2 Hybrid Binary INC/DEC design (Lookahead based) [36]

IV PROPOSED SYSTEM

Although efficiency has been described and studied, there is still an improvement in order to achieve a more efficient system. The basic Hybrid Binary INC/DEC design concept has previously been applied on two levels. However, we will construct a Multi-functional INC/Complement/Priority DEC/2's Encoder, a Modified Unified BCD/Binary Adder/Subtractor, a Floating Point Adder/Subtractor, a High Speed Multiplier, and Efficient Arithmetic Units for a Processor Core to make it more efficient.

V IMPLEMENTATION AND RESULTS

A. Multi-functional INC/DEC/2’s Complement/Priority Encoder:

The existing system uses decision signals with the least significant one bit of information (LSOB). Because increment, decrement, priority encoder, and 2's complement operations all need the discovery of LSOB, the decision block is shared by all of them. In Fig 1(b), the MUX-based binary INC/DEC circuit [2] has a decision block with N-1 (OR) gate delay. Fig 2 depicts an upgraded decision block with a delay of (N/8+9) gates (when 8-bit look ahead is used). This delay can be further lowered by employing OR gates with a prefix tree structure, resulting in a log2N OR gate delay.

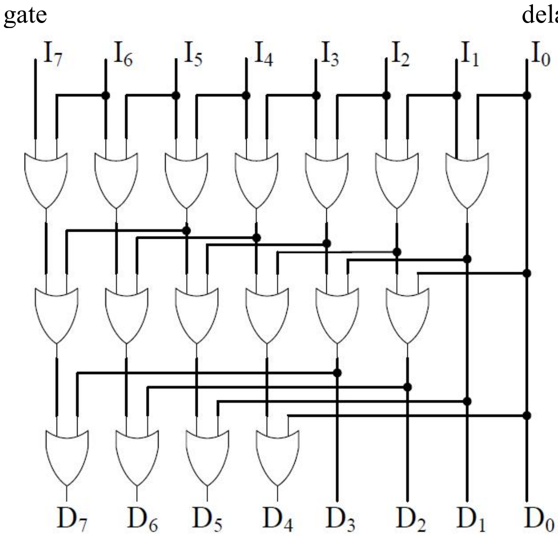

Figure 3 Prefix-Based Decision Block Type I

However, under the suggested arrangement, Fig 4 was the modified version of the structure in Fig 3. While the NAND/NOR gate requires only four transistors, the OR/AND gate requires six (extra 2 transistors for inverter). In addition, as compared to NOR/NAND gates, OR gates have an extra inverter (1 transistor) delay. As a result of the above implementation, the

ISSN : 2057-5688

area, power, and latency are minimised. When compared to the original design in Fig 3, the size of the structure in Fig 4 is reduced by 12 inverters, and the critical path delay is reduced by 2T (delay of 2 inverters).

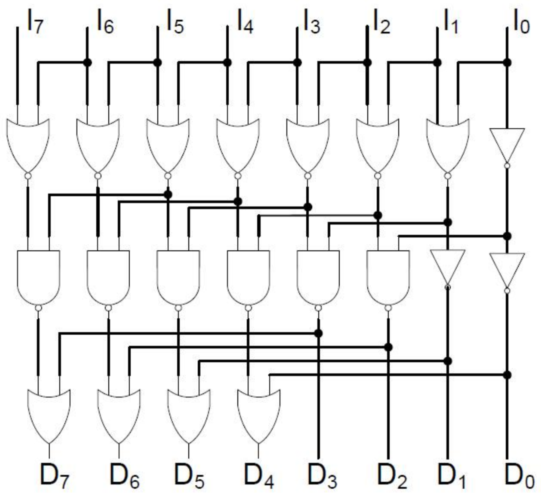

Figure 4 Prefix based decision block Type I with NOR-NAND

INC/DEC	Delay (nS)	Power (mW)	Power-Delay Product (pJ)	Area (um²)
Mux-based [2] (Fig 1(b))	5.640 (100%)	0.336 (100%)	1.895 (100%)	854 (100%)
Hybrid [3] (Fig 2)	2.386 (42.30%)	0.418 (124.40%)	0.997 (52.61%)	933 (109.25%)
With Proposed Decision Block Type I (Fig 3)	1.436 (25.46%)	0.414 (123.21%)	0.594 (31.35%)	1200 (140.51%)
With Proposed Delay Optimized Decision Block Type I (Fig 4)	1.221 (21.65%)	0.407 (121.13%)	0.497 (26.23%)	1137 (133.14%)

Table 1 Simulation results for 32-bit INC/DEC Circuits

B. A Modified Unified BCD/Binary Adder/Subtractor Architecture

As shown in Fig 5 the proposed unified BCD/Binary adder/subtractor design, which includes the pre-correction and post-correction stages. Whether the operation is binary or BCD is indicated by the 'Bin' signal. Bin = 1 denotes a binary operation, while Bin = 0 denotes a BCD operation[6]

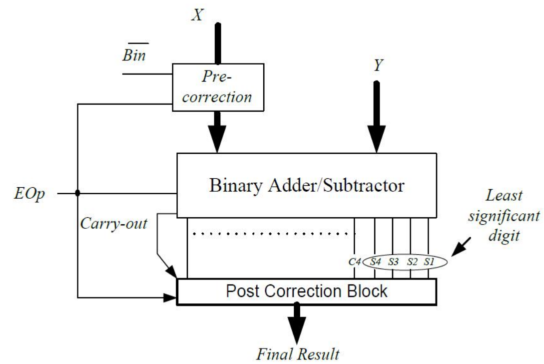

Figure 5 Architecture of unified BCD and binary adder / subtractor

Unlike Fischer's approach [4], the proposed architecture [6] does not use a complementing stage or a comparative stage, as does Humberto's approach [5]. The proposed architecture is contrasted to the Humberto architecture[5], which is the only unified adder/subtractor architecture that supports 2's complement signed, unsigned, and signed magnitude operands in this section. Because the Humberto architecture was built on an FPGA, both the suggested and Humberto architectures were constructed on an ASIC in this thesis for a fair comparison. The designs were structurally specified in Verilog HDL and simulated in Cadence Incisive Unified Simulator (IUS) v6.1 to cover all possible functional combinations. Using Cadence RTL Compiler v7.1, these architectures were mapped onto the TSMC 180nm technology typical library (operating circumstances of 1.8 V, 25oC). For determining the dynamic power, inputs were set to have a 50% toggle rate and a frequency of 1GHz. Table 2 shows how Humberto's architecture compares to the planned architecture. Because the suggested design eliminates the comparator and difficult pre-computation stage, it achieves a 13.6 percent latency improvement and a 14 percent area improvement. The proposed method can be extended to longer operand lengths, resulting in more efficient unified BCD/Binary adder / subtractor architectures.

	Humberto [53]	Proposed
Delay (nS)	4.004 (100%)	3.460 (86.4%)
Power (mW)	14.5 (100%)	13.37 (92.2%)
Power-Delay (pJ)	58.06 (100%)	46.26 (79.7%)
Area (um^2)	12068 (100%)	10498 (87%)

Table 2: Results for a 32-bit Unified BCD/Binary Adder/Subtractor

C. Comparator for 32-bit floating point unit

In an adder/subtractor circuit, a 2's complement block is necessary to rectify the 2's complement sum when the difference is negative. This correction, on the other hand, causes a log N increase in circuit latency and energy dissipation [7-10]. In Fig 5, a design methodology is developed and discussed that avoids the need for 2's complement circuits. This circuit is utilised in the proposed adder/subtractor to determine which of the operands is greater, and then a complement operation is performed using the end around carry approach. The proposed design's binary adder/subtractor structure, as seen in Fig 5, is depicted in Fig 6.

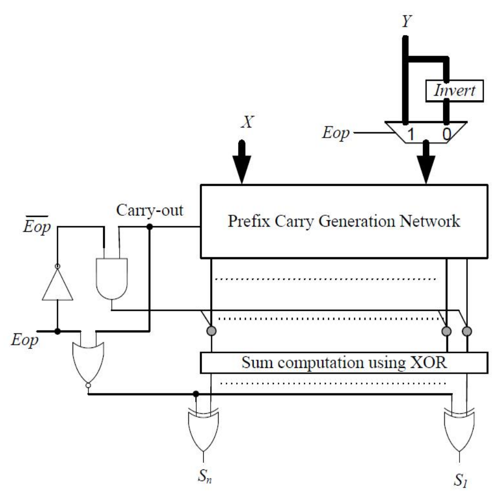

Figure 6 Implementation of Binary Adder/subtractor of Operands in signed magnitude form

An 8-bit comparator checks the exponents and an 8-bit subtractor calculates the amount by which the mantissas are to be shifted in a 32-bit single precision floating-point adder/subtractor unit. By inspecting the output Carry bit, a subtractor can also be utilised as a comparator, eliminating the need for a separate comparator circuit. If the subtractor's difference is negative, however, a 2's complement circuitry is necessary to fix the final result. This leads to a large reduction in critical path delay, as well as an increase in overall power and area. The floating point adder unit also has a leading zero anticipator (LZA) and a normalising unit in addition to the adder and comparator (which includes rounding off). For implementing an efficient floating point adder/subtractor unit, many optimised designs have been produced [8], and these optimised circuits are utilised in this study together with the proposed comparator and adder designs. All of the units were structurally specified in Verilog HDL and simulated in Cadence Incisive Unified Simulator (IUS) v6.1, which covered all possible functional combinations. Cadence RTL Compiler v7.1 was used to map these units onto the TSMC 180nm Technology typical library (operating conditions 1.8 V, 25oC). For estimating dynamic power, inputs were set to have a 50% toggle rate and a frequency of 1GHz. By combining the designs provided in the preceding chapters of this thesis, a complete 32-bit floating-point unit has been created. The results of comparing this proposed design [10] to existing floating point units [7-10] are shown in Table 3.

	Area (um^2)	Power (mW)	Delay (pS)	Power-Delay Product (pJ)
Existing Design	20073.2 (100%)	95437.249 (100%)	10560 (100%)	1007.81 (100%)
Proposed Design	17759.102 (88.47%)	86070.872 (90.19%)	9823 (93.20%)	845.47 (83.89%)

Table 3 A Comparison of Performance of Floating Point Adder Units

D. Implementation of High Speed Multiplier Design of Multipliers using Wallace and Dadda Algorithms

Wallace and Dadda were among the first to design and explain the use of special structures known as compressors and counters in multipliers for partial product reduction trees [11-12]. Compressors and counters are used to create a 16x16 bit multiplier partial product reduction tree in Fig 7 and Fig 8. Multi-operand addition can also be done with these counters/compressors .

ISSN : 2057-5688

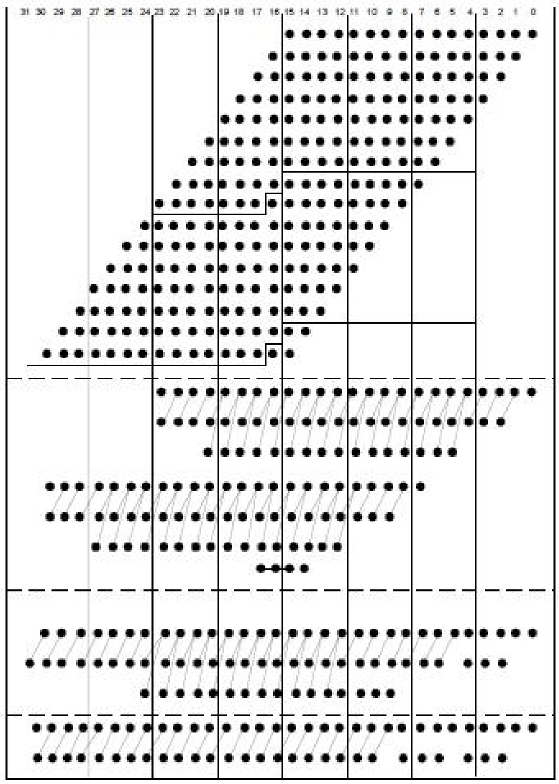

Figure 7 Wallace Algorithm for the design of a 16 bit Multiplier

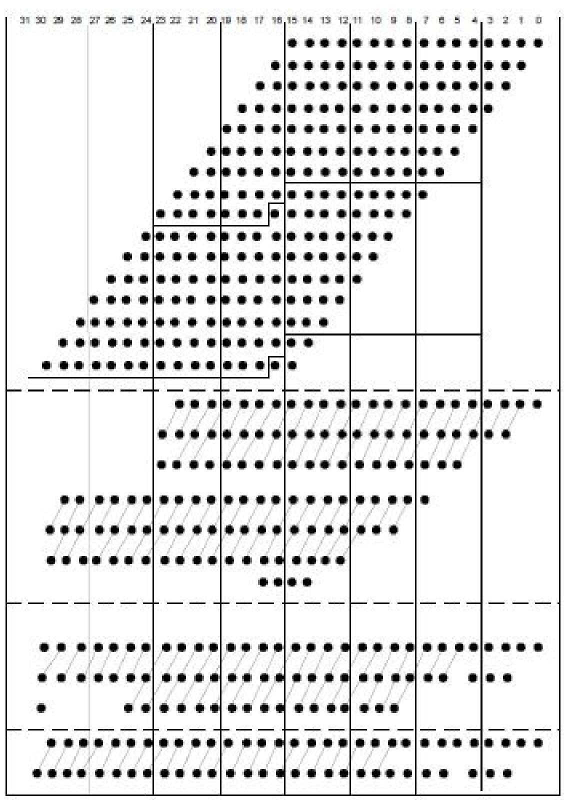

Figure 8 Dadda Algorithm for the design of a 16 bit Multiplier

Counters/compressors are usually followed by a high-speed adder in most multipliers. Multiplier critical route delays include not just the counter/compressor delay in the partial product reduction tree, but also the adder delay in the final step. The most important criterion for a high-performance multiplier, and hence an AL, is to design these units efficiently. It is apparent that these units are efficient based on the results of several compressors and counters that have been given. Counter-based designs have been shown to be more delay efficient than compressor-based designs [11-12], hence the multiplier in this section was conceived and implemented using counters. The counters/compressors are usually followed by a high-speed adder in most multipliers. Multiplier critical path delays include not only counter/compressor delays in the partial product reduction tree, but also adder delays in the final stage. The primary need for a high-performance multiplier, and hence an AL, is that these units be designed efficiently. These units are clearly efficient, as evidenced by the results of various compressors and counters that have been given. However, counter-based designs have been shown to be more delay-efficient than compressor-based designs [11-12], thus the multiplier in this section was conceived and built with counters.Table 4 shows that the power-delay product has improved significantly (by roughly 14.85 percent) as compared to the previous implementation.

	Area(um2)	Power (mW)	Delay(pS)	Power-Delay Product (pJ)
Existing Design	25771.33 (100%)	145.628 (100%)	11740 (100%)	1709.67 (100%)
Proposed Design	23461.61 (91.04%)	136.302 (93.60%)	10680 (90.97%)	1455.71 (85.15%)

Table 4 Simulation Results of a 32-bit Multiplier

E. Design of Efficient Arithmetic Block in an Arithmetic and Logic Unit (ALU)

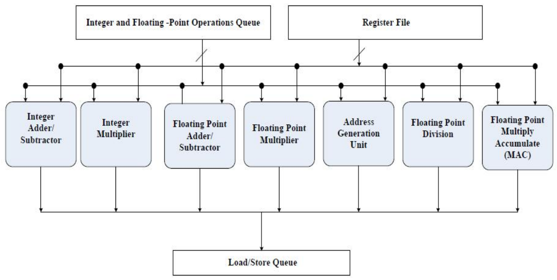

Figure 9 Microarchitecture of an Arithmetic unit in an AMD Processor Core

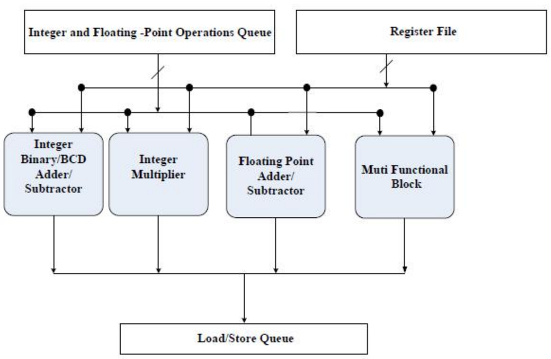

Figure 10 Processor Core with modifided functional units

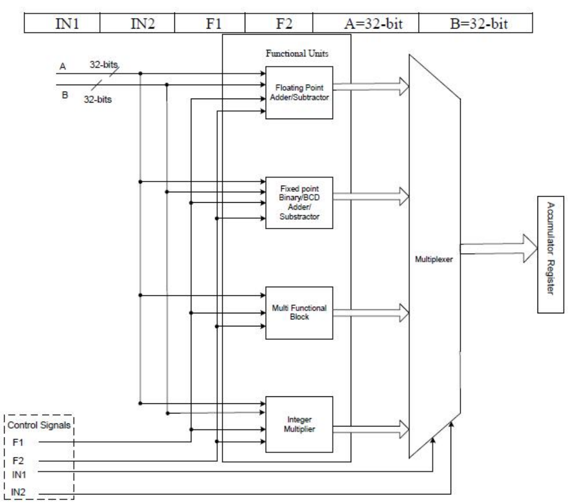

Figure 11 A generic architecture of an ALU

IN1	IN2	F1	F2	OPERATION
0	0	0	X	FLOATING POINT ADDITION
0	0	1	X	FLOATING POINT SUBSTRACTION
0	1	0	0	FIXED POINT ADDITION
0	1	0	1	FIXED POINT SUBSTRACTION
0	1	1	0	FIXED POINT BCD ADDITION
0	1	1	1	FIXED POINT BCD SUBSTRACTION
1	0	0	0	INCREMENTER
1	0	0	1	DECREMENTER
1	0	1	0	PRIORITY ENCODER
1	0	1	1	2'S COMPLIMENT
1	1	X	X	MULTIPLICATION

Table 5: Detailed list of operations

Simulation Results

	Area (um2)		Power (mW)		Delay (pS)		Power-Delay Product (pJ)	
	Existing	Proposed	Existing	Proposed	Existing	Proposed	Existing	Proposed
Control Unit and Bus Selection Unit	825 (100%)	825 (100%)	0.001 (100%)	0.001 (100%)	166 (100%)	166 (100%)	0.00017 (100%)	0.00017 (100%)
Floating point Adder/ Subtractor	20073.2 (100%)	17759.1 (88.47%)	95.437 (100%)	86.07 (90.12%)	10560 (100%)	9823 (93.20%)	1007.81 (100%)	845.47 (83.89%)
Fixed point Binary/BCD Adder/ Subtractor	12068 (100%)	10498 (86.99%)	14.5 (100%)	13.37 (92.21%)	4004 (100%)	3460 (86.41%)	58.06 (100%)	48.67 (83.83%)
Multi Functional Block	1432 (100%)	1637 (114.32%)	0.619 (100%)	0.624 (100.81%)	3497 (100%)	2333 (66.71%)	2.17 (100%)	1.46 (67.28%)
Fixed Point Multiplier	25771.3 (100%)	23461.6 (91.04%)	145.628 (100%)	136.302 (93.60%)	11740 (100%)	10680 (90.97%)	1709.67 (100%)	1455.71 (85.15%)

Table 12 Results of simulation results of ALU blocks

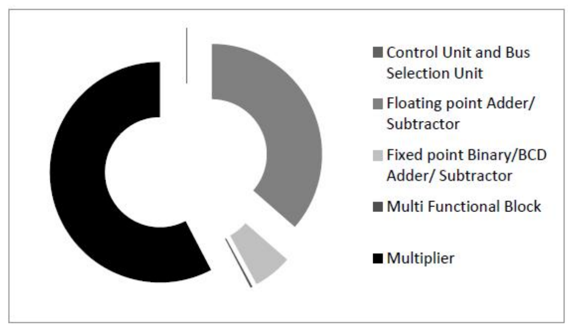

Figure 13 Power contribution of different arithmetic blocks in ALU

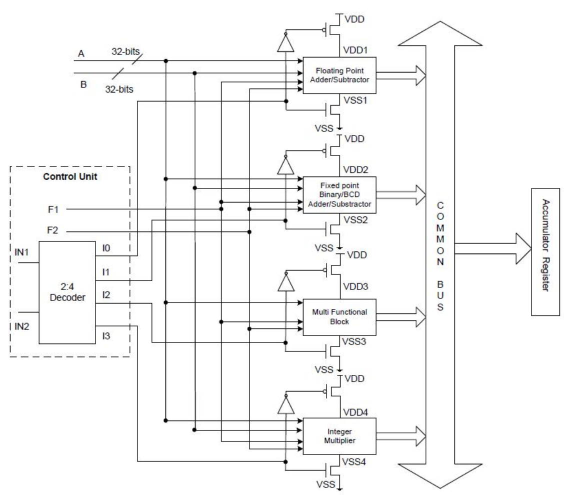

Figure 14 Arithmetic section of an ALU with power gating technique

Type	Power (mW)
Without Power-Gating Technique	236.367
With Power-Gating Technique	97.30

Table 7 Simulation Results for ALU while performing floating-point addition operation

VI CONCLUSION

This thesis focused on improving arithmetic circuits that, when combined, result in efficient recognition of an ALU's Arithmetic Unit. The thesis's initial contribution was the development of more efficient adder topologies that solve issues such as high fan-out, latency, and power consumption. While these systems have a delay overhead, they have the advantage of less fan-out and lower total energy consumption. Furthermore, efficient counter/compressor blocks have been devised to aid in the reduction of partial products in multipliers, resulting in efficient high-speed parallel multipliers. The second part of the thesis' contribution is the design of a multi-functional INCREMENT/DECREMENT/ 2's complement/ priority encoder circuit that has been proved to be efficient in terms of speed of operation without consuming excessive power. Because such a unit is so important in an ALU, including it results in more efficient arithmetic and logic units. Finally, all of the individual arithmetic units have been combined to implement the arithmetic element of an ALU, resulting in a circuit design that is more efficient than those previously published.

REFERENCES

[1] R. Hashemian and C. P. Chen. "A New Parallel Technique for Design of Decrement/Increment and Two's Complement Circuits." in Proceedings of the 34th Midwest Symposium on Circuits and Systems, volume 2, pages 887-890, 1991.

[2] Shaoqiang Bi, Wei Wang, and Asim Al-Khalili, "Multiplexer-based Binary Incrementer/decrementers," The 3rd International IEEE-NEWCAS Conference,19-22 June 2005. pp. 219-222.

[3] Veeramachaneni, S. Avinash, L. Kirthi, K.M. Srinivas, M.B. "A Novel High-Speed Binary and Gray Incrementer/Decrementer for an Address Generation Unit", International Conference on Industrial and Information Systems (ICIIS), 9-11 August 2007. pp.427-430.

[4] H. Fischer and W. Rohsaint. "Circuit Arrangement for Adding or Subtracting Operands Coded in BCD-Code or Binary-Code", Siemens Aktiengesellschaft, US patent 5146423, pages 1 – 9, Sep 1992.

[5] D.R.Humberto Calderón, G. N. Gaydadjiev, S. Vassiliadis, "Reconfigurable Universal Adder", Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP 07), pages 186-191, July 2007.

[6] Chetan Kumar, V. Sai Phaneendra, P. Ahmed, Syed Ershad, Veeramachaneni, Sreehari; Muthukrishnan, N. Moorthy; Srinivas, M.B.; , "A Unified Architecture for BCD and Binary Adder/Subtractor," Digital System Design (DSD), 2011 14th Euromicro Conference on , vol., no., pp.426-429, Aug. 31 2011-Sept. 2 2011.

[7] H. H. Saleh and E. E. Swartzlander, Jr., "A Floating-Point Fused Add-Subtract Unit," Proceedings of the 51st IEEE Midwest Symposium on Circuits and Systems, 2008, pp. 519 - 522.

[8] Jongwook Sohn, Earl E. Swartzlander Jr. "Improved Architectures for a Fused Floating-Point Add-Subtract Unit". IEEE Trans. on Circuits and Systems 59-I(10): 2285-2291 (2012)

[9] Jongwook Sohn "Improved architectures for a fused floating-point add-subtract unit", M.S. Thesis , The University of Texas at Austin,2011.

[10] H.H. Saleh, "Fused Floating-Point Arithmetic for DSP", PhD dissertation, Univ. of Texas, 2009.

[11] W.J. Townsend, E.E. SwartzlanderJr., and J.A. Abraham, "A Comparison of Dadda and Wallace Multiplier Delays," *Proc. SPIE,* Advanced Signal Processing Algorithms, Architectures, and Implementations XIII, pp. 552-560, 2003.

[12] Ron S. Waters, Earl E. Swartzlander, "A Reduced Complexity Wallace Multiplier Reduction", IEEE Transactions on Computers, vol. 59, no. 8, pp. 1134-1137, Aug. 2010.