Interval Arithmetic Logic Unit for Signal Processing and Control Applications

Ruchir Gupte, William Edmonson, Senanu Ocloo, Jaya Gianchandani, Winser Alexander.

23rd February 2006

North Carolina State University

1. Introduction
2. Architecture
3. Performance Metrics
4. Results
5. Future Work
6. Conclusion and Acknowledgements
7. References

Outline

1. **Introduction**
2. Architecture
3. Performance Metrics
4. Results
5. Future Work
6. Conclusion and Acknowledgements
7. References

Introduction

Our Purpose

- Solve problems in the signal processing field to obtain results with improved accuracy and at a faster rate.
- Build dedicated hardware to implement reliable computing in signal processing.

Path Followed

- Interval Arithmetic is one of the solutions to reduce errors resulting from numerical computations.
- Hence build an Interval ALU dedicated to signal processing and control applications.

Motivation

Problem

Rounding errors due to the use of Binary Floating Point Number System may be unacceptable in some signal processing applications.

Solution

Interval Methods can bound these errors that accrue due to Rounding.

We propose DSP specific Fixed Point Interval ALU as an improvement over existing Floating Point Interval ALUs.

- DSP Applications rarely require the full dynamic range offered by floating point number system.
- Floating Point Interval ALUs have the disadvantages of comparatively lower throughput, greater transistor count and being costlier.

Motivation

Problem

Rounding errors due to the use of Binary Floating Point Number System may be unacceptable in some signal processing applications.

Solution

Interval Methods can bound these errors that accrue due to Rounding.

We propose DSP specific Fixed Point Interval ALU as an improvement over existing Floating Point Interval ALUs.

- DSP Applications rarely require the full dynamic range offered by floating point number system.
- Floating Point Interval ALUs have the disadvantages of comparatively lower throughput, greater transistor count and being costlier.

Motivation

Problem

Rounding errors due to the use of Binary Floating Point Number System may be unacceptable in some signal processing applications.

Solution

Interval Methods can bound these errors that accrue due to Rounding.

We propose DSP specific Fixed Point Interval ALU as an improvement over existing Floating Point Interval ALUs.

- DSP Applications rarely require the full dynamic range offered by floating point number system.
- Floating Point Interval ALUs have the disadvantages of comparatively lower throughput, greater transistor count and being costlier.

Motivation

Problem

Rounding errors due to the use of Binary Floating Point Number System may be unacceptable in some signal processing applications.

Solution

Interval Methods can bound these errors that accrue due to Rounding.

We propose DSP specific Fixed Point Interval ALU as an improvement over existing Floating Point Interval ALUs.

- DSP Applications rarely require the full dynamic range offered by floating point number system.
- Floating Point Interval ALUs have the disadvantages of comparatively lower throughput, greater transistor count and being costlier.

Motivation

Problem

Rounding errors due to the use of Binary Floating Point Number System may be unacceptable in some signal processing applications.

Solution

Interval Methods can bound these errors that accrue due to Rounding.

We propose DSP specific Fixed Point Interval ALU as an improvement over existing Floating Point Interval ALUs.

- DSP Applications rarely require the full dynamic range offered by floating point number system.
- Floating Point Interval ALUs have the disadvantages of comparatively lower throughput, greater transistor count and being costlier.

Background

Implementation of Interval Algorithms in Software

Disadvantageous because of their slow speed for the following reasons:

- Function Calls.
- Memory Management.
- Changing Rounding modes.
- Exception Handling.

Variable Precision Interval Arithmetic Processors

Design by Michael J. Schulte and Earl E. Swartzlander.

- Uses the Floating Point Number System in its architecture.
- General Purpose ALU architecture.

Outline

1 Introduction

2 Architecture

3 Performance Metrics

4 Results

5 Future Work

6 Conclusion and Acknowledgements

7 References

Design Specifications

ALU Modules

Two Independent Modules operate in parallel to determine the lower bound and the upper bound of the output interval.

Accumulator

Accumulator in each of the modules to execute the “Multiply Accumulate” instruction efficiently.

Number Representation

All computations performed using 2’s complement fixed point arithmetic.

Rounding

- Design features a separate Rounding Unit.
- Choice of 24 or 16 bits at the output.

Design Specifications

ALU Modules

Two Independent Modules operate in parallel to determine the lower bound and the upper bound of the output interval.

Accumulator

Accumulator in each of the modules to execute the “Multiply Accumulate” instruction efficiently.

Number Representation

All computations performed using 2’s complement fixed point arithmetic.

Rounding

- Design features a separate Rounding Unit.
- Choice of 24 or 16 bits at the output.

Design Specifications

ALU Modules

Two Independent Modules operate in parallel to determine the lower bound and the upper bound of the output interval.

Accumulator

Accumulator in each of the modules to execute the “Multiply Accumulate” instruction efficiently.

Number Representation

All computations performed using 2’s complement fixed point arithmetic.

Rounding

- Design features a separate Rounding Unit.
- Choice of 24 or 16 bits at the output.

Design Specifications

ALU Modules

Two Independent Modules operate in parallel to determine the lower bound and the upper bound of the output interval.

Accumulator

Accumulator in each of the modules to execute the “Multiply Accumulate” instruction efficiently.

Number Representation

All computations performed using 2’s complement fixed point arithmetic.

Rounding

- Design features a separate Rounding Unit.
- Choice of 24 or 16 bits at the output.

Design Specifications

ALU Modules

Two Independent Modules operate in parallel to determine the lower bound and the upper bound of the output interval.

Accumulator

Accumulator in each of the modules to execute the “Multiply Accumulate” instruction efficiently.

Number Representation

All computations performed using 2’s complement fixed point arithmetic.

Rounding

- Design features a separate Rounding Unit.
- Choice of 24 or 16 bits at the output.

Top Level Block Diagram

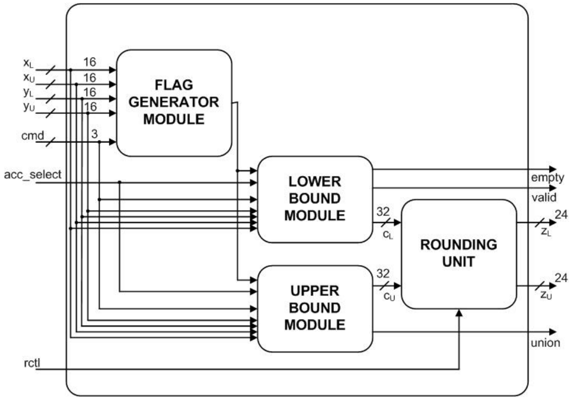
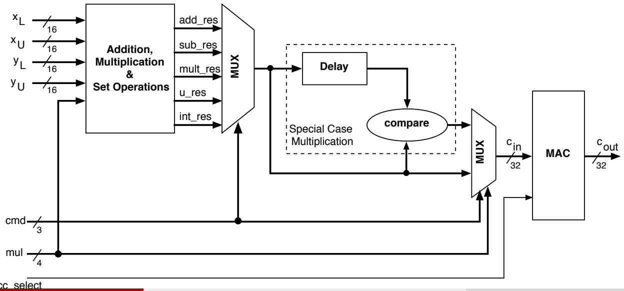


Figure: Top Level Block Diagram

Upper and Lower Bound Modules

- Both Modules operate in Parallel.
- Two clock cycles required for Special Case Multiplication.
- Two clock cycles required for finding the Union of disjoint intervals.
- One clock cycle required for all other operations.

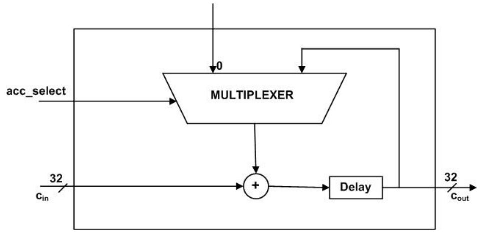
Figure: ALU Module



The Multiply Accumulate Design

- Extremely vital in DSP Applications to calculate the dot product.
  - $a.b = \sum_{i=0}^{n} a_i b_i$
- To be included as a part of the Instruction Set Architecture.

Figure: MAC Design



Arithmetic Operations performed by the ALU

- Addition.
  - $[x_L, x_U] + [y_L, y_U] = [x_L + y_L, x_U + y_U]$
- Subtraction.
  - $[x_L, x_U] - [y_L, y_U] = [x_L - y_U, x_U - y_L]$
- Multiplication.
  - 9 possible cases depending on input values relative to zero.
  - Important issue of Special Case Multiplication.
- Width.
  - $width[x_L, x_U] = x_U - x_L$
- Mid-Point.
  - $midpoint[x_L, x_U] = (x_U + x_L)/2$
  - Division by multiples of 2 is performed by shift operation.

Multiplication

Operation of the Flag Generator Module

Value	Case	Result
0001	$x_L \geq 0; y_L \geq 0$	$[x_L y_L, x_U y_U]$
0010	$x_L \geq 0; y_L < 0 < y_U$	$[x_U y_L, x_U y_U]$
0011	$x_L \geq 0; y_U \leq 0$	$[x_U y_L, x_L y_U]$
0100	$x_L < 0 < x_U; y_L \geq 0$	$[x_L y_U, x_U y_U]$
0101	$x_L < 0 < x_U; y_U \leq 0$	$[x_U y_L, x_L y_L]$
0110	$x_U \leq 0; y_L \geq 0$	$[x_L y_U, x_U y_L]$
0111	$x_U \leq 0; y_L < 0 < y_U$	$[x_L y_U, x_L y_L]$
1000	$x_U \leq 0; y_U \leq 0$	$[x_U y_U, x_L y_U]$
0000	$x_L < 0 < x_U; y_L < 0 < y_U$	$[\min(x_U y_L, x_L y_U), \max(x_L y_L, x_U y_U)]$

Logical Operations performed by the ALU

Set Union

- $[x_L, x_U] \cup [y_L, y_U] = [min(x_L, y_L), max(x_U, y_U)]$
- Union of two disjoint sets results in two intervals.
  - union$[x_L, x_U] \cup [y_L, y_U] = [min(x_L, y_L), min(x_U, y_U)] + [max(x_L, y_L), max(x_U, y_U)]$

Set Intersection

- $[x_L, x_U] \cap [y_L, y_U] = [max(x_L, y_L), min(x_U, y_U)]$
- Intersection of two disjoint sets results in a null set.

Logical Operations performed by the ALU

Set Union

- $[x_L, x_U] \cup [y_L, y_U] = [min(x_L, y_L), max(x_U, y_U)]$
- Union of two disjoint sets results in two intervals.
    - union$[x_L, x_U] \cup [y_L, y_U] =$ $[min(x_L, y_L), min(x_U, y_U)] + [max(x_L, y_L), max(x_U, y_U)]$

Set Intersection

- $[x_L, x_U] \cap [y_L, y_U] = [max(x_L, y_L), min(x_U, y_U)]$
- Intersection of two disjoint sets results in a null set.

Rounding

Characteristics of Outward Rounding

- Lower bound of the resulting interval rounded to negative infinity.
  - Implemented by discarding the bits of lower significance.
- Upper bound of the resulting interval rounded to positive infinity.
  - Implemented by adding the 'OR'ed value of the lower significance bits to be discarded.
- Significance of using 2's complement number representation.
  - Algorithm holds true for positive and negative numbers.

Block Diagram for Rounding

Lower Bound Rounding

Upper Bound Rounding

Figure: Lower Bound

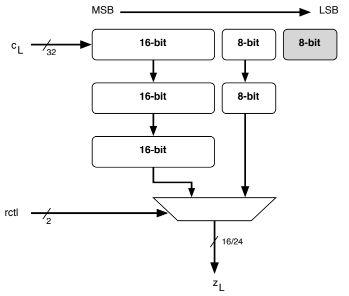
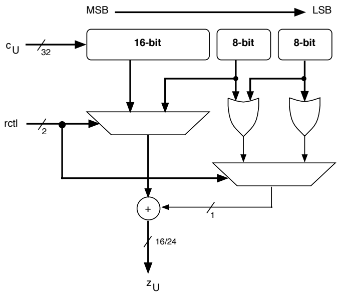


Figure: Upper Bound



Outline

1 Introduction

2 Architecture

3 **Performance Metrics**

4 Results

5 Future Work

6 Conclusion and Acknowledgements

7 References

Determining the Performance of the Design

Throughput

- Most critical to achieve high throughput for DSP applications.
- One of the design goals is to obtain higher throughput than the floating point architecture.

Design Area

- Optimize the design to minimize the on-chip area.

Power Consumption

- Make circuit modifications which minimize power consumption.
- Power analysis of prime importance for space applications.

Numerical Reliability

- Verify the functionality of the design for all input combinations.

Determining the Performance of the Design

Throughput

- Most critical to achieve high throughput for DSP applications.
- One of the design goals is to obtain higher throughput than the floating point architecture.

Design Area

- Optimize the design to minimize the on-chip area.

Power Consumption

- Make circuit modifications which minimize power consumption.
- Power analysis of prime importance for space applications.

Numerical Reliability

- Verify the functionality of the design for all input combinations.

Determining the Performance of the Design

Throughput

- Most critical to achieve high throughput for DSP applications.
- One of the design goals is to obtain higher throughput than the floating point architecture.

Design Area

- Optimize the design to minimize the on-chip area.

Power Consumption

- Make circuit modifications which minimize power consumption.
- Power analysis of prime importance for space applications.

Numerical Reliability

- Verify the functionality of the design for all input combinations.

Determining the Performance of the Design

Throughput

- Most critical to achieve high throughput for DSP applications.
- One of the design goals is to obtain higher throughput than the floating point architecture.

Design Area

- Optimize the design to minimize the on-chip area.

Power Consumption

- Make circuit modifications which minimize power consumption.
- Power analysis of prime importance for space applications.

Numerical Reliability

- Verify the functionality of the design for all input combinations.

Outline

1 Introduction

2 Architecture

3 Performance Metrics

4 **Results**

5 Future Work

6 Conclusion and Acknowledgements

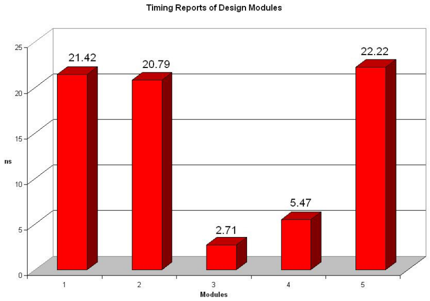
7 References

Execution Times

Timing Analysis of Various Modules

Module Name	Timing
Lower Bound	21.42 ns
Upper Bound	20.79 ns
Left Round	2.71 ns
Right Round	5.47 ns
Design	22.22 ns

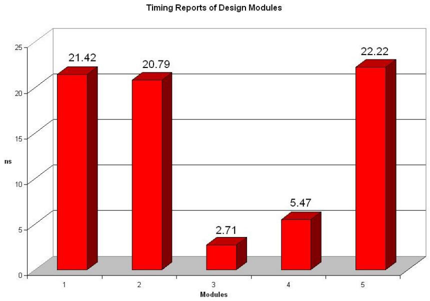
Design Clock Frequency = 45 MHz.



Execution Times

Timing Analysis of Various Modules

Module Name	Timing
Lower Bound	21.42 ns
Upper Bound	20.79 ns
Left Round	2.71 ns
Right Round	5.47 ns
Design	22.22 ns

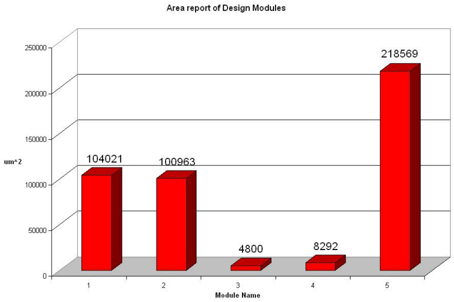


Design Clock Frequency = 45 MHz.

Design Area

Area of Various Modules

Module Name	Area (μm^2)
Lower Bound	104021
Upper Bound	100963
Left Round	4800
Right Round	8292
Design	218569

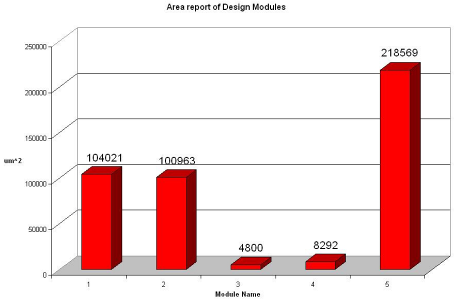


Technology Used: 0.18μm CMOSX Library

Design Area

Area of Various Modules

Module Name	Area (μm^2)
Lower Bound	104021
Upper Bound	100963
Left Round	4800
Right Round	8292
Design	218569



Technology Used: 0.18μm CMOSX Library

Outline

1. Introduction
2. Architecture
3. Performance Metrics
4. Results
5. **Future Work**
6. Conclusion and Acknowledgements
7. References

Future Work on the Design

Expand hardware to evaluate the following functions:

- Logarithm.
- Trigonometry.
- Exponential.

Pipeline Architecture

- Restructure the design with pipeline multipliers and adders.
- Improve the most critical metric,“Throughput”.

Power Analysis

- Run Power scripts to get power estimates of the design.
- Make the design modular to minimize power consumption.

Outline

1 Introduction

2 Architecture

3 Performance Metrics

4 Results

5 Future Work

6 **Conclusion and Acknowledgements**

7 References

Conclusions

- Interval Arithmetic as a solution to errors resulting from numerical computations.
- Advantages of Fixed Point I-ALUs over Floating Point I-ALUs for DSP Applications.
- Hardware solutions to software implementation of Interval Methods.
- Design specifications of our architecture.
  - Overall architecture.
  - Introduction of dedicated hardware to execute the MAC instruction.
  - Outward Rounding to eliminate precision errors.
- Throughput and area analysis.
- Improvement in the design to increase throughput.

Acknowledgements

Thesis Advisor

- Dr. William Edmonson, Associate Professor, NC State University.

Graduate Students

- Senanu Ocloo, Ph.D Candidate, NC State University.
- Jaya Gianchandani, NC State University.

- Dr. Winser Alexander, Professor, NC State University.

Outline

1. Introduction
2. Architecture
3. Performance Metrics
4. Results
5. Future Work
6. Conclusion and Acknowledgements
7. **References**

References

- ANSI/IEEE, IEEE Standard for Binary Floating-Point Arithmetic. New York: ANSI/IEEE Std 754-1985, 1985.
- R. E. Moore, Interval Analysis. Prentice-Hall, Inc., 1966.
- E. Hansen, Global Optimization Using Interval Analysis. Marcel Dekker, Inc., 1992.
- U. Kulisch, Advanced Arithmetic for the Digital Computer. New York: Springer-Verlag, 2002.
- M. Schulte and E. Swartzlander, “A Family of Variable-Precision Interval Arithmetic Processors”. IEEE Transactions on Computers, Vol 49, No. 5, May 2000.
- R. Kolla, A. Vodopivec, J. Wolff v. Gudenberg, “The IAX Architecture Interval Arithmetic Extension.”, April 1999.

Thank You !!