RAVIOLI–Reconfigurable Arithmetic Variable-precision Implementations of On-Line Instructions

Robert McIlhenny
Computer Science Department
California State University, Northridge
Northridge, CA 91330
rmcilhen@csun.edu

Miloš D. Ercegovac
Computer Science Department
Univesity of California, Los Angeles
Los Angeles, CA 90095
milos@cs.ucla.edu

Abstract

In this paper, we present a library of floating-point arithmetic operators that can be dynamically reconfigured for either real-number or complex-number mode, and for which the precision is variable and is set prior to compilation. The library is compared to corresponding modules built using the Xilinx Alliance CORE [3] library of floating-point arithmetic operators for the implementation of a unit that generates the inverse of a matrix. A significant lower cost and total cycle delay is demonstrated.

1. Introduction

Reconfigurable arithmetic is a class of arithmetic operations that can change mode to perform either a real number or a complex number operation, based on switching the functionality of the underlying components. The functionality is determined at run time, based on setting the value associated with a particular flag. This is accomplished through programming both modes onto a chip and using internal multiplexers to select either real or complex mode. This allows flexibility to support either mode after the chip has been programmed, and allows reuse of hardware.

Several authors have investigated reconfigurable arithmetic, specifically regarding real/complex number multiplication. Barazesh, et. al. [2] developed a VLSI signal processor with complex number arithmetic capability, based on a multimode architecture that realizes either a real or complex number multiplication by means of pipelining stages. Aoki, et. al. [1] presented the design of a reconfigurable parallel multiplier that realizes (i) single-precision complex number multiplication; (ii) double-precision real number multiplication; and (iii) a pair of single-precision real number four-operand multiply-add operations.

In this paper, a library of arithmetic operators which can be dynamically reconfigured is presented. Support for variable-precision exponents is achieved through implementing simple binary adders, subtractors, and shifters based on bit-wide adders and bit-wide registers. Since the size of such components is linear with precision, extension is easily achieved. Support for variable-precision mantissas requires implementations that are linear with precision. Traditionally, assuming precision n, floating-point adders and subtractors have size $O(n)$, but floating-point multipliers, dividers, and square root units have size $O(n^2)$. To achieve $O(n)$ size, on-line arithmetic is adopted, using a redundant signed-digit representation of operands. This require modules to convert from standard binary to redundant signed-digit representation, and vice versa. An overview of on-line arithmetic is presented next.

2. On-line arithmetic

On-line arithmetic is a class of arithmetic operations in which all operations are performed digit serially, in a most significant digit first (MSDF) manner. One of the key components of on-line arithmetic is the on-line delay δ defined as the number of operand digits necessary to generate the first digit of the result. Each successive digit is generated one per cycle. The total latency, assuming m-digit precision is $\delta + m - 1$.

The inherent digit-pipeline characteristic provides the means for efficient hardware implementations. On-line arithmetic has several advantages, including: (i) the ability to overlap dependent operations, since output digits are produces serially, most-significant digit first, enabling successive operations to begin before previous operations have completed, and (ii) support for variable precision, since once a desired precision is obtained, successive outputs can be ignored.

Proceedings of the 13th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'05)
0-7695-2445-1/05 $20.00 © 2005 IEEE

3. Library of modules

The library of reconfigurable arithmetic modules can be set for precision at compile time and can be set for radix based on a flag rc ($rc = 1$ for complex radix $2j$ mode, $rc = 0$ for real radix 4 mode). The input (I) and output (O) ports for each module, and associated port bit-widths are shown in Table 1, for input operands *in0* and *in1* and output operand *out0*.

Port	Width	I/O	Description
clk	1	I	clock for internal flip-flops
reset	1	I	reset for internal flip-flops
rc	1	I	radix mode flag
enable_in0	1	I	enable input in0
enable in1	1	I	enable input in1
exp_in0	e	I	exponent of input in0
exp_in1	e	I	exponent of input in1
man_in0	4	I	mantissa of input in0
man_in1	4	I	mantissa of input in1
enable_out0	1	O	enable output out0
exp_out0	e	O	exponent of output out0
man_out0	4	O	mantissa of output out0

Table 1. Module input and output ports

The modules were coded in VHDL, implemented using Xilinx Foundation 6.1, and individually mapped to a Virtex-II Pro XC2VP7-7 FPGA. Parameters include the cost in terms of CLB slices for general precision m radix-4 borrow-save digit mantissas (each digit consisting of 4 bits), and e-bit exponents, and the on-line delay δ.

Modules for implementing radix $2j/4$ on-line floating-point arithmetic instructions include: addition (Ol2j4FloatAdd), multiplication (Ol2j4FloatMult), divsion (Ol2j4FloatDiv), and square root (Ol2j4FloatSqrt). These are shown in Table 2.

Module name	Cost (CLB slices)	On-line delay δ
Ol2j4FloatAdd	$2.5m + 4e + 78$	3
Ol2j4FloatMult	$22m + 3e + 56$	9
Ol2j4FloatDiv	$32.5m + 3e + 169$	9
Ol2j4FloatSqrt	$23.5m + 3e + 134$	3

Table 2. Library of modules

4. Application

We apply the RAVIOLI library of reconfigurable floating-point arithmetic modules toward the implementation of a unit that generates the inverse of a 2×2 matrix which has either real elements or complex elements. Given a matrix

$$M = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad (1)$$

the inverse matrix is

$$M^{-1} = \begin{bmatrix} \frac{d}{ad-bc} & \frac{-b}{ad-bc} \\ \frac{-c}{ad-bc} & \frac{a}{ad-bc} \end{bmatrix} \quad (2)$$

To generate M^{-1} requires two multiplications and one subtraction to produce the denominator (or equivalently the determinant of matrix M), and four divisions to produce the elements of M^{-1}. We compare the design to a corresponding design using networks of the Xilinx Alliance CORE real-number floating-point arithmetic operators [3], where complex operators are constructed as networks of real operators. The cost results in terms of total Virtex CLB slices, and throughput results in terms of computed bits per cycles, are shown in Table 3, for precisions of 24-bit mantissas (both real and imaginary components) and 8-bit exponents. For the implementation using the RAVIOLI modules, the 24-bit binary real and imaginary components of the mantissa are converted to/from a unified 24-digit radix 2j/4 mantissa by conversion units before/after matrix inversion computation.

Design	Cost (CLB slices)	Throughput (bits/cycle)
Alliance CORE	16216	0.67
RAVIOLI	3892	1.09

Table 3. Results for matrix inversion

References

[1] T. Aoki, H. Amada, and T. Higuchi. Real/complex reconfigurable arithmetic using redundant complex number systems. *Proceedings of the 13th symposium on computer arithmetic*, pages 200–206, 1997.

[2] B. Barazesh, J. Michalina, and A. Picco. A vlsi signal processor with complex arithmetic capability. *IEEE transactions on circuits and systems*, 35-5:495–505, May 1988.

[3] Xilinx. AllianceCORE floating point cores. October 2004.

Proceedings of the 13th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'05)
0-7695-2445-1/05 $20.00 © 2005 **IEEE**