Microelectronics Journal 49 (2016) 10–18

Contents lists available at ScienceDirect

Microelectronics Journal

journal homepage: www.elsevier.com/locate/mejo

An efficient multiple precision floating-point Multiply-Add Fused unit

K. Manolopoulos [a], D. Reisis [a,*], V.A. Chouliaras [b]

[a] Department of Physics, Electronics Laboratory, National Kapodistrian University of Athens, Greece
[b] Department of Electronics and Electrical Engineering, Loughborough University, UK

ARTICLE INFO

Article history:
Received 24 March 2014
Received in revised form
5 August 2015
Accepted 16 October 2015
Available online 4 January 2016

Keywords:
Floating-point
Multiply-Add Fused
Multiple precision
VLSI

ABSTRACT

Multiply-Add Fused (MAF) units play a key role in the processor's performance for a variety of applications. The objective of this paper is to present a multi-functional, multiple precision floating-point Multiply-Add Fused (MAF) unit. The proposed MAF is reconfigurable and able to execute a quadruple precision MAF instruction, or two double precision instructions, or four single precision instructions in parallel. The MAF architecture features a dual-path organization reducing the latency of the floating-point add (FADD) instruction and utilizes the minimum number of operating components to keep the area low. The proposed MAF design was implemented on a 65 nm silicon process achieving a maximum operating frequency of 293.5 MHz at 381 mW power.

1. Introduction

Widely used applications such as graphics, transforms, digital filters, to name a few, impose the need for Floating-Point calculations, which consequently are a feature of all high performance computers, digital signal processors (DSPs) and graphic accelerators [1–5]. Moreover, applications in the area of 3D graphics require the parallel execution of floating point operations, which usually involve operands with single and/or double precision format. To meet these demands, a significant number of novel processor designs provides floating-point units (FPUs), which support not only single but also double precision operations.

While the two most common floating-point formats, namely single and double precision, are satisfactory for the vast majority of computational applications, there are a few emerging applications where they are not adequate. Examples have been identified in the fields of climate modeling and computational physics [6–9], among others, that require even higher precision. To support such applications, IEEE has now included a standard for 128-bit ("quad") precision as part of its IEEE 754-2008 Standard for Floating-Point Arithmetic [10]. Quadruple precision computations can be handled either by software or hardware, but hardware solutions are more effective in terms of performance. Most of the aforementioned applications involve algorithms utilizing the $(A \times B)+C$ single-instruction equation, which is realized in general purpose processors (GPPs) by combining a floating-point multiplier and a floating-point adder into a Multiply-Add Fused (MAF) unit. Due to the importance of MAF instructions, many GPPs include embedded MAF units to speed-up the execution of SIMD instructions, while others use them to replace entirely the floating-point adder and multiplier units [11–13].

Furthermore, the importance of combining high precision floating point operations and the MAF instruction can be assessed by the efforts of major manufacturers towards providing commercially available solutions. Notable examples are first, the AMD "Bulldozer" microarchitecture which provides two 128-bit MAF cores per module, supporting single, double and extended precision format [14]; second, the IBM z13 microprocessor which performs MAF instructions for single, double and quadruple precision formats [15].

Aiming at improving the MAF performance, this paper presents a multi-functional, multiple precision MAF architecture based on a dual-path organization, which operates efficiently in quadruple, double or single precision.

The proposed design can perform one quadruple precision MAF operation, or two double precision operations in parallel, or four single precision operations in parallel. Moreover, it can also perform – in each precision mode – multiple stand-alone multiply or add operations.

The motivation for this work was first, to present a MAF architecture supporting a variety of functions without significantly increasing area and delay; second, to investigate how the dual-path algorithm can efficiently be used to provide a MAF design with multiple functionalities; and third, to evaluate its performance by identifying the hardware cost and the critical path delays. For the sake of comparison, this paper also presents the synthesis results of implementing four MAF architectures by using the same technology with the proposed MAF: first, a typical double precision MAF

* Corresponding to: Department of Physics, Electronics Laboratory, National Kapodistrian University of Athens, Panepistimiopolis, Physics Buildings IV & V, Athens 15784, Greece. Tel.: +30 2107276720; fax: +30 2107276801.
E-mail address: dreisis@phys.uoa.gr (D. Reisis).

http://dx.doi.org/10.1016/j.mejo.2015.10.012
0026-2692/

architecture; second, a dual-mode, dual-path MAF unit that operates in both double and single precision; third, a typical quadruple precision; and fourth, a multi-block design including a typical quadruple precision MAF and two dual-mode dual-path MAF units.

The paper is organized as follows: Section 2 reviews the related work in the area of floating-point Multiply-Add Fused units. Section 3 describes the structure of a conventional MAF unit. Section 4 presents the architecture of the proposed multiple precision floating-point MAF unit and describes in detail the basic modules of the MAF design. Section 5 shows the area and delay results and finally, Section 6 concludes the paper.

2. Related work

The Fused Multiply-Add unit has been introduced in the IBM POWER1 processor (1990), also known as RS/6000 [16]. Since then, several designs have been proposed and realized in an effort to improve the efficiency of the MAF architecture. The authors of [17] compare the design complexity of single or dual-pass (through the multiplier array) when realizing a fused multiply-add floating-point unit. In [18] a MAF floating-point unit with signed digit addition is presented: a signed digit addition along with a two step normalization method reduces the latency of the addition. Bruguera and Lang [19] present a floating-point MAF unit that computes floating-point addition with lower latency than floating-point multiplication and MAF. This is accomplished by bypassing, in case of an addition, the pipeline stages related to the multiplication. Sun and Gao [20] introduce a two data-path MAF floating-point unit; and based on whether a subtraction occurs, it activates only one of the paths to reduce the average latency. He and Li [21] demonstrate a MAF floating-point unit able to process denormalized numbers at the cost of slightly increased hardware complexity and operation latency. Li and Li [22] present a fully pipelined MAF unit combining the final addition with rounding. In [23] a bridged MAF design is presented. This architecture includes a common floating-point multiplier and adder and by adding specific hardware components between them, creates a “bridge” sending data from the multiplier unit to the adder in order to perform the MAF instruction. More complex architectures are presented in [24] and [25]: Huang et al. [24] proposes a dual-mode floating-point MAF unit with the SIMD feature executing instructions of either double or single precision and Gok and Ozbilen [25] describe a multi-functional double and quadruple precision floating-point MAF unit.

In addition to the above MAF designs there are several other architectures presented in the literature, which involve stand-alone floating-point multipliers and adders. Notable examples of relevant research are published in [26–29]. The authors in [26] present a floating-point multiplier executing addition and rounding in parallel. In [27] a dual-precision multiplier is proposed. It utilizes half-sized multiplication arrays to perform a double precision multiplication in three cycles or a single precision multiplication in two cycles. Akkas and Schulte [28] presents two designs performing dual-mode floating-point multiplication. The first architecture executes either one quad-precision multiplication or two double precision multiplications in parallel. The second design performs one double precision multiplication or two single precision multiplications in parallel. In [29] two architectures performing dual-mode floating-point addition are presented. The first is based on the single-path algorithm [30] and supports a double precision or two single precision additions in parallel. The second design implements the two path algorithm [30] and performs either a quad-precision addition or two double precision additions in parallel.

Compared to the literature above, our proposed architecture is the only design, which provides triple functionality allowing it to operate in all the precision modes, while being capable of performing multiple operations.

3. Conventional MAF unit

This section highlights the IEEE floating-point standard format [10] and a typical Multiply-Add Fused unit. According to the IEEE standard for binary floating-point arithmetic a floating-point number requires three fields: sign S, biased exponent E and fraction F. Fig. 1 depicts the fields of the different precision formats and the following equation shows the value X of a normalized floating-point number:

$$X = (-1)^S \times 2^{E-bias} \times 1.F$$

A brief description of a typical double precision MAF architecture (Fig. 2) included in several general purpose processors [17,18, 21, 22, 24], is given in the following paragraphs.

The IEEE representation of a double precision floating point number requires 1-bit as a sign, 11-bit for the biased exponent and 53-bit fraction. The execution of a MAF instruction is accomplished through a number of serial steps, that are categorized in three stages:

1. *Multiplication stage*: the first stage multiplies the 53-bit significands of the operands A and B. At the same time, the third operand C is inverted and aligned. This is performed in parallel with the multiplication, in order to reduce the latency, with the addend C positioned 56 bits left of the product. Next, the exponent difference is calculated: $diff = expC - (expA + expB - 1023)$ where expA, expB, expC are the biased exponents of the operands A, B and C respectively and 1023 is the bias for IEEE754 double precision arithmetic. Hence, the necessary shift amount equals:

   $$sh = 56 - diff \Rightarrow sh = expA + expB - expC - 967$$

   Then, the alignment is performed as a right-shift by the 161-bit aligner.
2. *Addition stage*: the second stage adds the 106-bit product $A \times B$ and the 161-bit aligned operand C by using a 106-bit 3:2 CSA and a 161-bit adder. In parallel with the addition, the Leading Zero Anticipator (LZA) unit processes the lower 106-bits of the 3:2 CSA and it computes the necessary shift amount for normalization. If it is required, the adder result will be complemented before forwarding the result to the last stage.
3. *Normalization and rounding*: the third stage performs normalization and rounding to produce the final result. The 161-bit result is normalized based on the estimation of the LZA unit. Next, rounding is performed and, if required, a post-normalization. At the same time, the sign and the exponent of the final result are prepared.

4. Proposed architecture overview

The motivation behind this work has been to research and design an efficient MAF architecture able to operate in different precision modes while being capable of performing multiple instructions in an SIMD fashion. We follow a design approach leading to an architecture with high performance and reduced latency, at relatively modest hardware cost. We have designed the MAF unit to support three different precision modes: quadruple, double and single precision. In each of these modes the design performs a number of different operations: MAF instruction, stand-alone multiplication, or stand-alone addition.