FPMax: a 106GFLOPS/W at 217GFLOPS/mm² Single-Precision FPU, and a 43.7GFLOPS/W at 74.6GFLOPS/mm² Double-Precision FPU, in 28nm UTBB FDSOI

Jing Pu¹, Sameh Galal², Xuan Yang¹, Ofer Shacham³, Mark Horowitz¹

¹Stanford University, Stanford, CA USA, ²Soft Machines Inc., Santa Clara, CA USA, ³Google Inc., Mountain View, CA USA

Abstract

FPMax implements four FPUs optimized for latency or throughput workloads in two precisions, fabricated in 28nm UTBB FDSOI. Each unit’s parameters, e.g. pipeline stages, bootstrap encoding etc., were optimized to yield 1.42ns latency at 110GLOPS/W (SP) and 1.39ns latency at 36GFLOPS/W (DP). At 100% activity, body-bias control improves the energy efficiency by about 20%; at 10% activity this saving is almost 2x. Keywords: FPU, energy efficiency, hardware generator, SOI

Introduction

Floating-point (FP) computation has become ubiquitous in digital systems for either sequential or parallel workloads. To help designers create efficient implementations for these different environments, we created FPGen, an FPU generator, which extracted design innovations from 50 years of published research on FPU design [1]. FPGen explores different implementation techniques to find those that optimize the design for the desired applications’ power, performance and area constraints. The FPMax chip evaluates the capability of FPGen and Ultra-Thin Body BOX fully-depleted SOI (UTBB FDSOI) technology by incorporating four generated FP Multiply-Accumulate (FMAC) units optimized for different precisions and either latency or throughput applications in ST 28nm UTBB FDSOI LVT technology.

All 4 FPUs are fully pipelined, implement IEEE compliant rounding, and utilize internal forwarding before rounding [8]. They use widely different implementations for the FMAC (Table I). The designs optimized to minimize the latency use a cascade multiply-add (CMA) architecture with a Wallace tree to sum up the partial products (PPs). The throughput optimized designs use a fused multiply accumulation (FMA) design with simpler combiners for the multiplication.

FPU Architectures

For latency oriented FPUs, the primary metrics are the energy/FLOP and the average latency per FLOP. In FMAs [2], the latencies for using the result as a multiplier or an adder input are the same (Fig. 1(a)). However, in many applications, accumulation dependencies tend to be more common. A CMA input are the same (Fig. 1(a)). However, in many applications, accumulation dependencies tend to be more common. A CMA input are the same (Fig. 1(a)). However, in many applications, accumulation dependencies tend to be more common. A CMA

For GPU type applications with abundant parallelism, the latency of an individual operation is less critical. As a result, the metrics become the energy per FLOP and the compute efficiency in GFLOPS/mm² [2]. We find that FMAs are more area efficient than CMAs. The focus on area and energy efficiencies again leads to the use of Booth 3 encoding, and also simpler combiner structures for the multiplier partial products: the DP units uses a simple array and the SP uses a modified array called a ZM structure [3].

Chip Implementation and Measurement

The design parameters of the FPUs were selected from the Pareto curves of energy vs. performance shown in Fig. 3 for SP throughput designs. The curve with triangle marks represents the performance for designs with different architectural parameters simulated at 1V supply using FPGen. Given the ability to change $V_{DD}$, the fabricated SP FMA performance is illustrated by the curve marked with white squares. In addition, adding the body-bias (BB) control of the UTBB FDSOI process improves energy efficiency (at a constant area) by 21%, or improves area efficiency (at a constant energy) by 20%.
This design can achieve 289GFLOPS/W at 79 GFLOPS/mm² in low energy mode, and 278GFLOPS/mm² at 60GFLOPS/W in high performance mode. The measurement of the DP throughput unit, DP FMA, is also shown in Fig. 3. Using VDD and BB it achieves 117GFLOPS/W at 13GFLOPS/mm² in low energy mode, and 111GFLOPS/mm² at 20GFLOPS/W.

Fig. 4 provides energy vs. performance tradeoffs for the latency optimized designs. The performance metric is the average delay, which is the product of the clock period and the average cycles per FLOP (i.e. one plus the average latency penalty) when running the SPEC FP benchmarks. The BB control provides energy reduction in two ways. First, it reduces the power by approximately 13% if the unit is heavily used, by lowering VDD and Vt. The problem with this statically set BB is that our FPU generator creates working designs which match numbers better than actual silicon. These results demonstrate that our FPU generator creates working designs which match its performance estimates, allowing designers to quickly create designs optimized for their application. It also demonstrates the advantages of strong Vt control, since this allows one to save 20% of the energy when compute bound, but not lose 3x in leakage when the unit is marginally utilized.

The measurement results were obtained using a built-in test capability as shown in Fig. 5. High speed on-chip RAMs are implemented to feed/store the inputs/outputs of the selected FPU during a test run (at full FPU speed). A JTAG interface is used to load and check values in the RAMs in a lower speed.

| Power | Area (mm²) | Frequency (MHz) | Peak GFLOPS | Energy Efficiency (GFLOPS/W) | Energy Efficiency (GFLOPS/mm²) |
|-------|-----------|-----------------|------------|-------------------------------|-------------------------------|
| 66mW  | 0.032     | 1.19GHz         | 289        | 314                           | 278                           |
| 41mW  | 0.024     | 910MHz          | 278        | 314                           | 278                           |
| 25mW  | 0.018     | 910MHz          | 278        | 314                           | 278                           |
| 17mW  | 0.0081    | 910MHz          | 278        | 314                           | 278                           |

**TABLE II. PERFORMANCE COMPARISON**

**References**

[1] S. Galal, et al., *IEEE Symp. Computer Arithmetic*, pp. 25-34, 2013.
[2] S. Galal, et al, *IEEE Trans. on Computers*, vol. 60, pp. 913-922, 2011.
[3] D. Zuras, et al, *J. Solid-State Circuits*, vol. 21, pp. 814–819, 1986.
[4] H. Kaul, et al., *JSSCC Dig. Tech. Papers*, pp. 182-184, 2012.
[5] J. Kao, et al., *IEEE Asian Solid State Circuits Conf.*, pp. 1–4, 2010.
[6] H. Oh, et al., *J. Solid-State Circuits*, vol. 41, pp. 759-771, 2006.
[7] S. Jain, et al., *Int. Conf. VLSI Design*, pp. 252-257, 2010.
[8] S. Trong, et al., *IEEE Symp. Computer Arithmetic*, pp. 77-86, 2007.