ARM Floating-Point 2016

David Lutz

Four areas of effort/concern

- Full support for binary16 arithmetic
- Faster implementations of basic functions
- Coping with wire delay
- High-Precision Anchored (HPA) accumulators

Binary16

- Comes from graphics (especially colors)
- Two times the computation per memory access
- Perform any operation that exists in binary32 or binary64
- Subnormal support more important

Faster implementation of basic functions

- FMA has decreased from 9 cycles (5 mul, 4 add) to 6 cycles (3 mul, 3 add) to less
- Divide has gone from 2 bits/cycle to 4 bits/cycle to more
  - FMA-based divides: not currently a good idea
- Converts are now twice as fast (2 cycle latency)
- How long will latency continue to improve?
- Splitting FMAs into separate Mul/Add functions still gives better performance

Typical 4-cycle FMA

- all 3 operands needed at the beginning of the operation
- sum of 4 products: s = a*x + b*y + c*z + d*w

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
fmul s,a,x	M	M	M	M												
fma s,b,y					F	F	F	F								
fma s,c,z									F	F	F	F				
fma s,d,w													F	F	F	F

ARM 6-cycle FMA with separate multiply and add

- 3-cycle multiply followed by 3-cycle add
- Note that a single FMA is slower
- sum of 4 products: s = a*x + b*y + c*z + d*w

	1	2	3	4	5	6	7	8	9	10	11	12	13
fmul s,a,x	M1	M2	M3										
fma s,b,y		M1	M2	M3	A1	A2	A3						
fma s,c,z					M1	M2	M3	A1	A2	A3			
fma s,d,w								M1	M2	M3	A1	A2	A3

Wire delay hits synthesized datapath

- Less logic depth = less delay?
- Booth vs. non-Booth multiplier array
- AND/OR reduction (counting set bits 0, 1, many)

High Precision Anchored Accumulators

- Problem: non-associativity of FP addition (both correctness and reproducibility)
  - Kulisch solution not implementable (area), uses too much power
  - Software solutions too slow?
- Problem insight: we don't need all of FP range for most sums
  - Galaxies and subatomics don't mix
  - Most sums at national labs have <128 bits of range
- Proposed solution: HPA number (long integer, anchor)
  - The anchor specifies the smallest magnitude we care about
  - The long integer is interpreted with respect to the anchor
  - Fully associative if we don't overflow

Implementing HPA

- Long integer addition - easy extension to existing 128- or 256-bit SIMD units
- Anchor and FP input broadcast to all lanes
- At least as fast as non-associative FP addition
- Sum of products
- All sums associative until converted back to FP
- National labs are interested in software model

Conclusions

- New tool: binary16
- Binary FP implementation is not yet optimal
- Wires are changing these designs
- High interest in associative accumulation, HPA one possible solution