

Soft FP Unit: A Data-Flow Perspective

Adam Covington
Brandon Heller
Chip Kastner

Today's discussion

- How can a data-flow processor be used to implement a soft FP unit?
- What are the data-flow characteristics of our IXP FP implementation?
- What IXP architecture changes will enable the unit to act more like a data-flow processor?

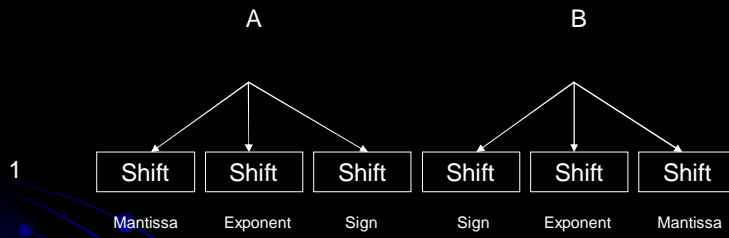
Defining the scope

- Let's assume that, instead of an IXP, we implemented our soft FP unit on a single data flow processor.
- For simplicity, we'll assume this data flow processor has an instruction set identical to that of an IXP microengine.
- How would a soft floating-point unit work?

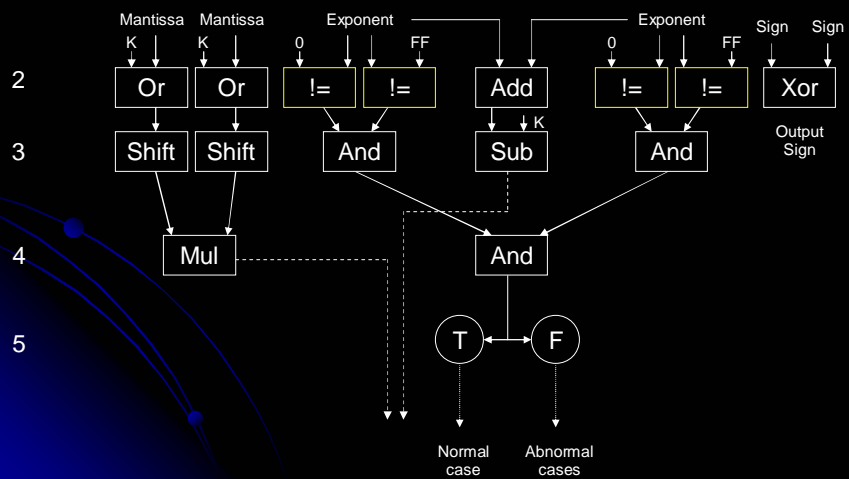
Inside a floating-point operation

- The data flow processor seeks to maximize concurrent execution of instructions
- There is plenty of concurrency available within the code for floating-point operations
- With the microengines, we were limited to performing instructions serially

Unpacking



Decision Making



Speedup

- The original code, executed serially, takes 17 instruction cycles
- With enough operation units (8) and fanout available, it takes 5 cycles
- Over 3x speedup for this portion!
- Of course, utilization of 8 operation units would never be 100%

Compared with IXP

- Latency would be much lower
- With the IXP, we would be able to run multiple FP operations concurrently
- There would be much less communication overhead here
- Potential exists for higher throughput

Another thought:

How our FP unit follows
data-flow principles

Data Flow and our Soft FPU

- One could say our FPU has some data flow characteristics
 - Operation units = Microengines
 - We use a similar instruction packet as a data flow processor: An instruction and two operands
 - Microengines, like operation units, sit idle but begin processing as soon as instruction packet arrives

Data Flow and our Soft FPU

- Arbitration network is similar to our SRAM pull bus + SHaC unit
 - Routes instruction packets to specified operation units (microengines)
- Distribution network is similar to our SRAM push bus + SHaC unit
 - Operands distributed back to instruction cells (our server processor)

Differences from a true data-flow processor

- Implicit operation unit destination in instruction packets
- Implicit signaling
- No queuing on the SHaC
- All of the above translates into lower latency

Architectural Changes for a Data-Flow Based IXP

- Replace SHaC queue with a switching network to enable faster instruction packet transmission
- Hardware to immediately perform operation when operands present
- Ability to have fanout ≥ 2 (operand multicasting) would be beneficial
- Would each microengine be a data-flow processor, or the entire IXP?