The FirePath™ LIW Processor and ISA Extensions for Broadband

Sophie Wilson

Technical Director

Broadband Carrier Access

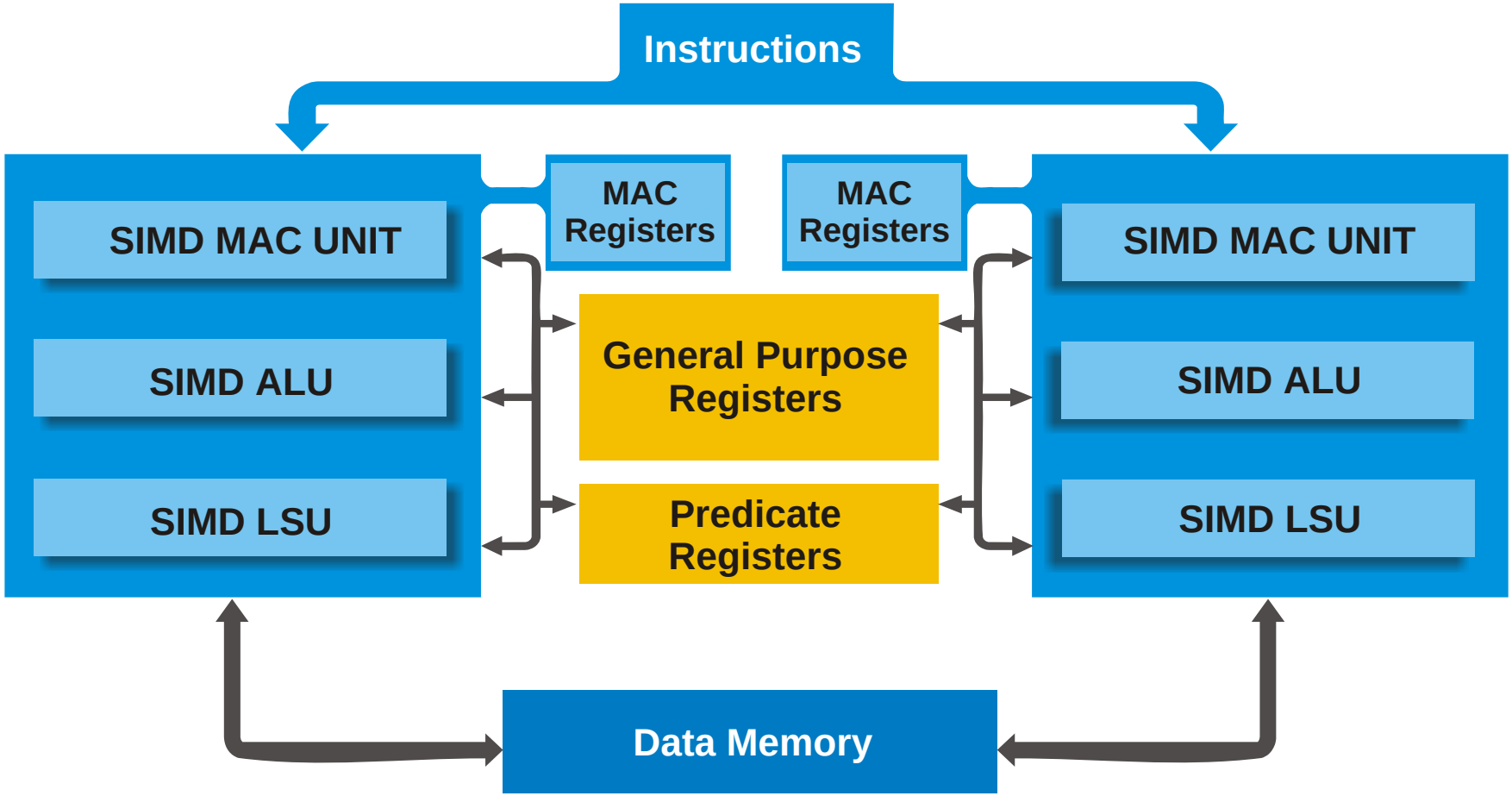
FirePath History

- Element 14 formed 26th July 1999
- Element 14 acquired by Broadcom Nov 2000
- Using FirePath initially as a processor core
- First FirePath SoC targetted at xDSL
- Now in two generations of Central Office chips
  - This talk describes changes from the first to the second generation
  - Firepath is now starting to be used in Voice over IP at the Central Office

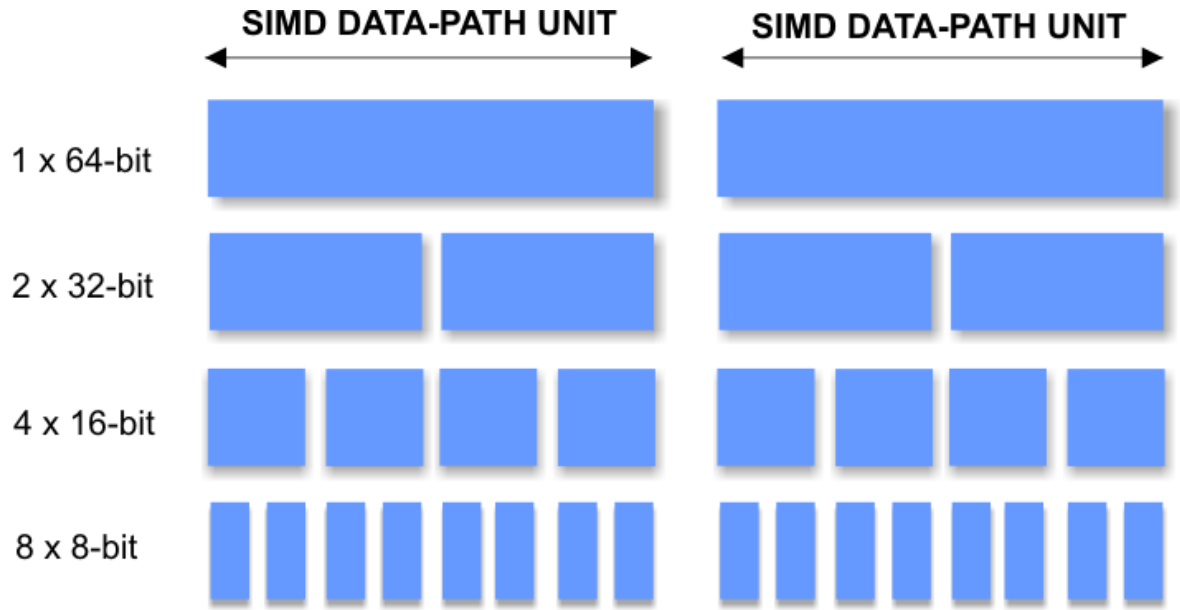
Key FirePath architecture targets

- "Bulk Data Processing"
  - Churn through big volume of data, applying the same (complex) processing to all of it.
- Support for particular applications
  - Communications: signed half-words, words
  - Audio/Compressed Voice: signed half-words, words
  - Video: signed and unsigned bytes
- Cope with lots of different algorithms for the data

FirePath Processor



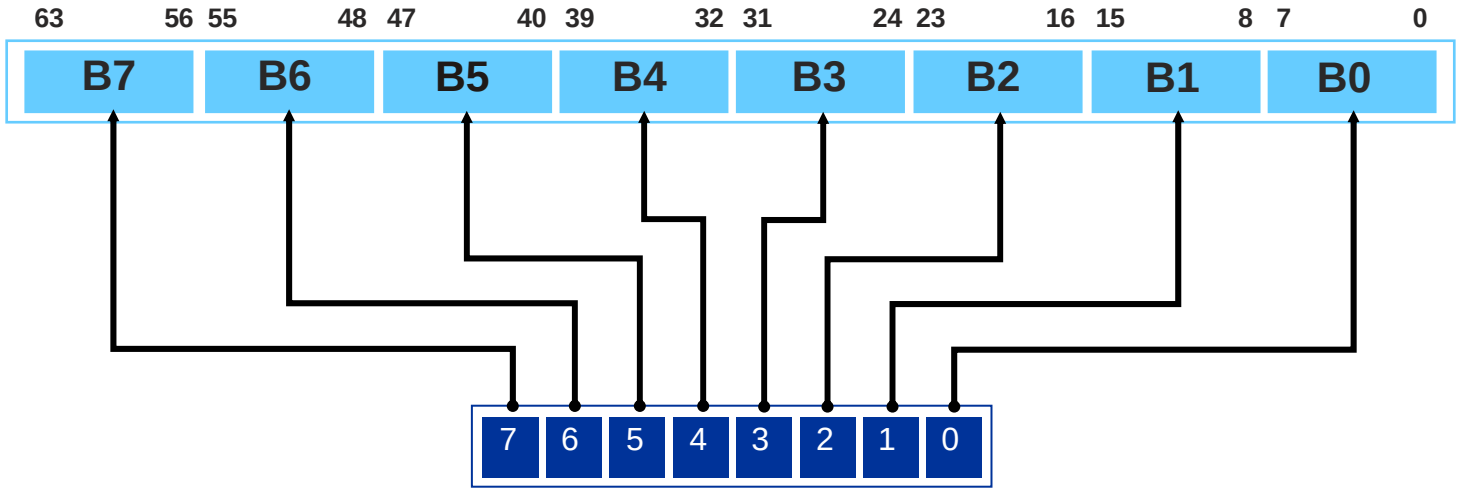
FirePath Parallelism



- Two forms of parallelism:
  - LIW: two identical 64-bit data-paths (compiler friendly)
  - SIMD: each data-path is SIMD-laned

SIMD Predication

- Predicate bit per byte lane:



8 bit predicate register

FirePath Architecture: LIW

- 128-bit execution width via LIW/SIMD machine
  - Instruction level parallelism exploited via two symmetric 64-bit RISC pipelines with 64 common 64 bit registers
  - Data parallelism exploited via 2, 4 or 8-way SIMD within each RISC pipeline

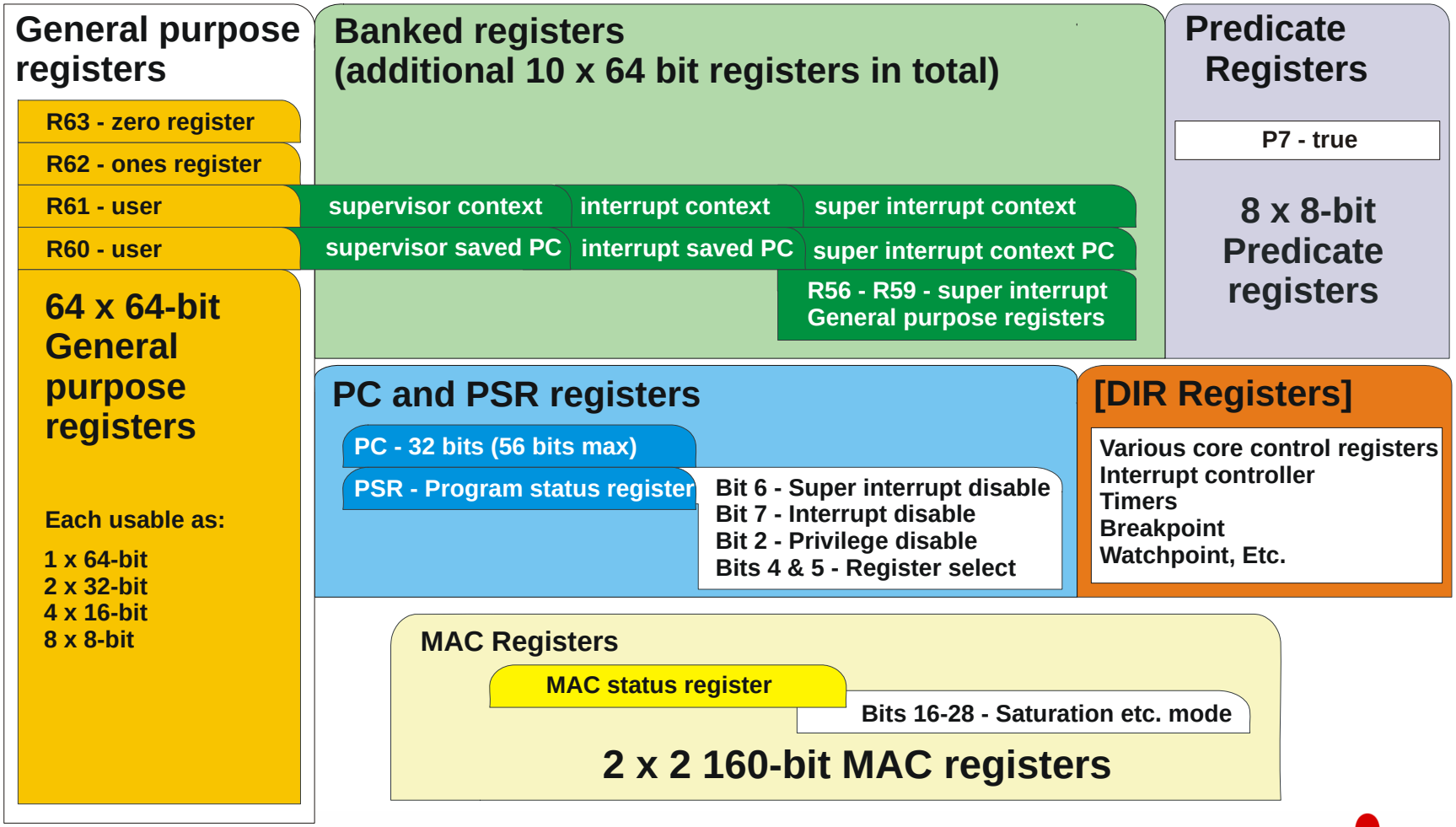
FirePath Architecture: ISA

- Instruction set
  - Complete and largely orthogonal SIMD set
  - Supports DSP and control code
  - Specific support for communications algorithms such as Galois field arithmetic

FirePath Architecture: Compute

- Accessible compute power
  - Large general purpose register file (64 off 64-bit registers)
  - Integrated multiply-accumulate (MAC) unit with dedicated accumulator file
  - Per clock tick, can perform and sustain for example:
    8x16-bit load operations (load 128 bits),
    8x16-bit MAC operations,
    1 address pointer update

FirePath Architectural State



Successful chips

- BCM6410 CO DSL products shipping in volume
  - Three generations since 2000 (FP2000, FP2001, FP2002)
- BCM6411 CO ADSL2+ in volume shipment
  - Instruction set extended (FP2003)
- BCM6510 CO ADSL2+/VDSL2 volume ramping
- BCM6513 CO VoIP in evaluation

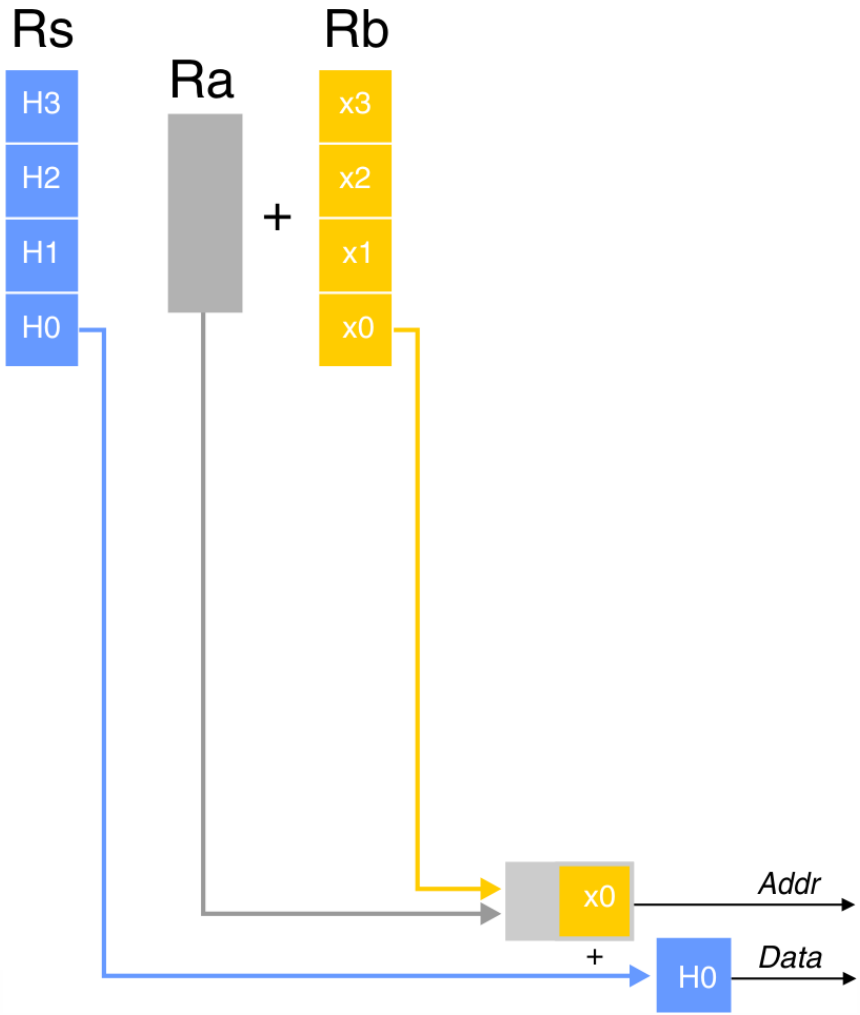
Second Generation Requirements

- Better, faster, cheaper
  - But without Moore's law to help us – 0.13 again
- More compute required in the same space!
- "Point Accelerations"
  - But we already have application specific support ☹

Architecture and Point Accelerations

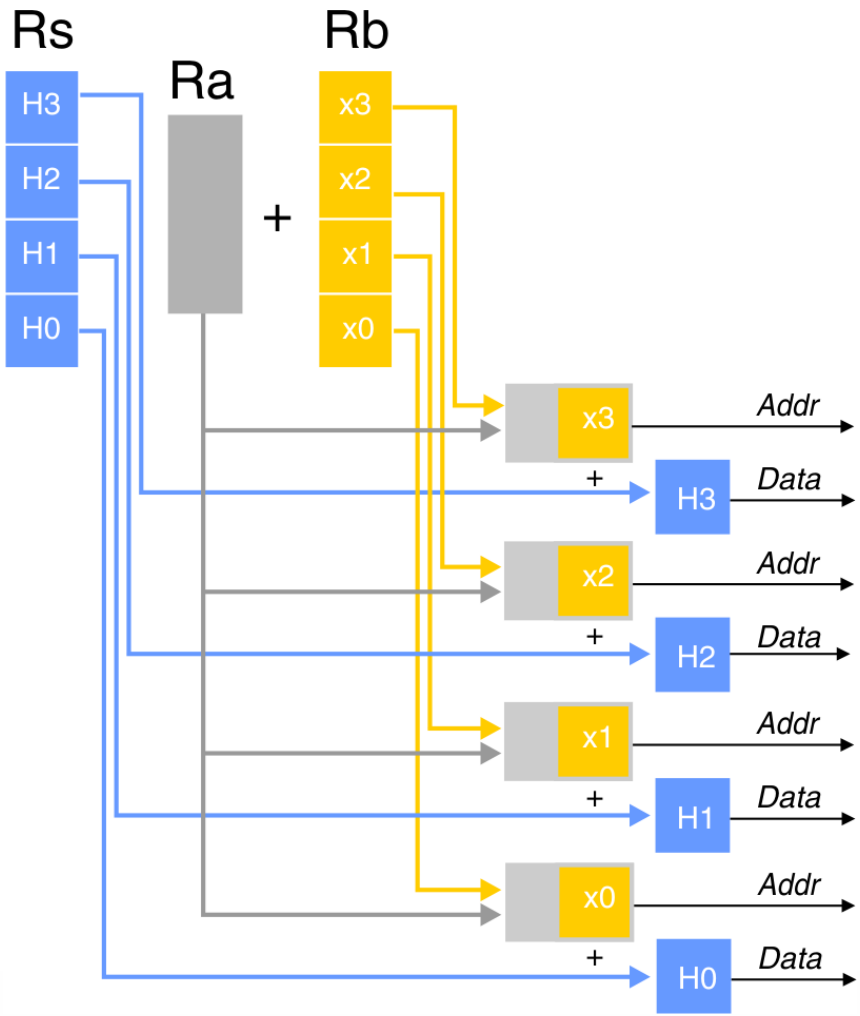
- FirePath architecture opens new possibilities
  - 64 bit SIMD values
  - Wider issue capability
    - Three Reads per side per cycle
  - Wider (and asynchronous) result capability
    - Two writes per side per cycle
  - LIW
    - Can use both sides in one go

A Simple SIMD Point Acceleration



- First generation already had an STH Rs,[Ra,Rb]
  - One half word written to computed address

ST4HX



- Second generation adds ST4HX Rs,[Ra,Rb]
  - Four half words written to computed address using Rs and Rb as SIMD values
- Only small speed improvements
  - No change to memory bandwidth: one cycle per store

Viterbi Acceleration (first generation)

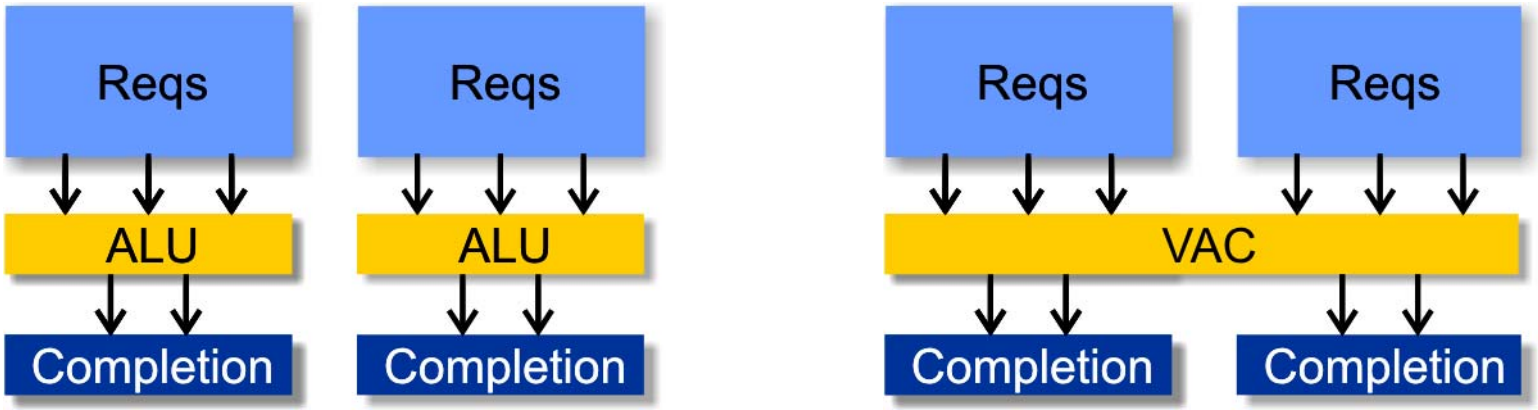
- Already have (SIMD) Galois field operations
  - MULG, MACG, SUMG, ADDB, CMPB
  - 8 way SIMD parallel (per side)
- Viterbi decoder
  - Parallel algorithm devised for original FirePath
  - SIMD predication very useful
- Greater parallelism needed
  - No simple replacement gives enough gain

Viterbi Acceleration (second generation)

- Use a single instruction spanning both sides
  - Minor change to issue part of machine required
  - 32 way SIMD parallel
- A very ugly instruction indeed…
  - VAC accout0/accout1, accout2/accout3, accin0/accin1, accin2/accin3, del0, del1
- Limits to use – can't issue one per cycle

Viterbi Acceleration (second generation)

- Change to microarchitecture



Viterbi Acceleration (new machine)

- Limits to use – can't issue one per cycle
  - Can overcome this by careful writing of the software
- Large overall saving: 14% of total cycles in the most extreme case
  - This was already an accelerated application with many different phases: a 14% gain is huge!

Categories of Point Accelerations

- Added too many things to cover them all at once!
- New load/stores
  - Six new load/store operations (like ST4HX)
- Viterbi operations
  - 10 new operations (VAC and its support instructions)
- Multiply Pipe operations
  - 5 new operations
- DSL Specific operations
  - 7 DSL specific operations

Conclusions

- You can always speed things up
- The machine architecture has a big effect on what you can do
- The machine microarchitecture also has a big effect on what you can do
- You need to enlist your programmers, too