How SIMD Width Affects Energy Efficiency: A Case Study on Sorting

Hiroshi Inoue
IBM Research – Tokyo
Goal & Approach

Goal:
- to understand how SIMD width affects execution time and energy consumption
  - *Not* to propose a new energy-efficient algorithm or system

Approach:
- to take SIMD mergesort as an example
- to measure **execution time**, **power** and **energy** (= execution time × power) with various hardware configurations on a commodity PC
  - SIMD width (8-way AVX, 4-way SSE or 1-way scalar)
  - Memory bandwidth
SIMD mergesort

- Combining advantages of sorting networks (SIMD friendly) and usual mergesort (lower computational complexity)
  - usual comparison-based mergesort in memory
    • computational complexity of $O(N \log(N))$
    • mostly sequential memory accesses
  - vector-register-level bitonic merge operation implemented with SIMD min/max instructions
    • data parallelism
    • less conditional branch

$\Rightarrow$ Wider vector gives sub-linear reduction in the number of instructions
SIMD-based merge for values in two vector registers

Input
- Two vector registers contain four presorted values in each

SIMD merging
- One SIMD comparison and “shuffle” operations for each stage without conditional branch

Output
- Eight values in two vector registers are now sorted

(example of bitonic merge)
Evaluation

- **Hardware:** a commodity PC + external power meter
  - Core i7 4770 (Haswell) 3.4 GHz, 4 cores, 8 threads
  - one or two 4-GB DDR3-1333 DIMMs (single or dual channel)
  - power meter Yokogawa WT-210 (for system-level power)
  - Redhat Enterprise Linux 6.5, gcc-5.2

- **Tested algorithms (for sorting random 256-M 32-bit integers)**
  - SIMD mergesort w/ scalar (1 way), SSE (4 way), or AVX (8 way)
  - radix sort (scalar)
  - quicksort (std::sort, scalar)
Summary of observations

1. **Execution time**
   - Wider SIMD gives larger speedup (up to 10x)

2. **Power**
   - SIMD increases power only up to 15%

3. **Energy** ($= \text{Execution time} \times \text{Power}$)
   - Lower energy consumption with wider SIMD

4. **Power** and **Execution time** with lower bandwidth-to-compute ratios
   - Wider SIMD may yield better performance with lower power!

Refer to paper (not covered today)
- Energy consumption with various bandwidth-to-compute ratios (achieved using DVFS)
  - Need to balance core compute performance and memory bandwidth to minimize energy consumption
Execution time (scalar vs. SIMD with 1 thread)

SIMD mergesort

- 9.7x speedup by 8-way SIMD
- 6.8x speedup by 4-way SIMD

Wider SIMD gave larger speedup as expected
Execution time (scalar vs. SIMD with 8 thread)

- 5.0x speedup by 8-way SIMD
- 4.4x speedup by 4-way SIMD

Smaller gains from SIMD due to memory bandwidth bottleneck
Execution time (8-way vs. 4-way)

- Mergesort with 8-way SIMD: 14% speedup
- Mergesort with 4-way SIMD: 42% speedup
- Mergesort without SIMD: 84% speedup
- Radix sort without SIMD: 96% speedup

8-way SIMD (AVX) gave additional speedups over 4-way SIMD (SSE)
Increase in power by use of SIMD was not so significant
Energy consumption was significantly reduced due to shorter execution time.
Energy ($= \text{Execution time} \times \text{Power}$) with 8 threads

- Mergesort with 8-way SIMD: 4.6x reduction in energy by 8-way SIMD
- Mergesort with 4-way SIMD: 3.9x reduction in energy by 4-way SIMD

Energy consumption was significantly reduced due to shorter execution time.
Energy \( (= \text{Execution time} \times \text{Power}) \) 8-way vs. 4-way

- **mergesort with 8-way SIMD**: 38% less energy with 3% higher power
- **mergesort with 4-way SIMD**: 16% less energy
- **mergesort without SIMD**: 42% less execution time with 3% higher power
- **quicksort (std::sort)**: 14% less execution time with 2% lower power
- **radix sort without SIMD**: better

✓ Wider SIMD yielded better performance with lower power when using 8 threads
Wider SIMD yields better performance with lower power when using 8 threads.
Power and Execution time with reduced bandwidth

![Graph showing the relationship between throughput (1/sec) and power (watt) for different merge sort algorithms with and without SIMD, with 2 memory channels (full bandwidth) and 1 memory channel (half bandwidth). The graph indicates that wider SIMD yields shorter time and lower power.]

- With lower memory bandwidth, power reduction by SIMD was more significant.

Wider SIMD yields shorter time and lower power.

Better (lower power)
Summary & Future work

- Summary of this study
  - Wider SIMD gives larger speedup and less energy consumption
  - Also, it potentially yields lower power by reducing number of instructions when bandwidth-to-compute ratio is low
  - (It is important to balance core performance and memory bandwidth to achieve best energy efficiency)

Increasing SIMD width will be important for future low-power processors even with limited bandwidth-to-compute ratios

- Future work
  - to evaluate with other workloads, especially floating-point intensive applications