Empirical Study of Power Consumption of x86-64 Instruction Decoder

Mikael Hirki, Zhonghong Ou, **Kashif Nizam Khan ,** Jukka K. Nurminen, Tapio Niemi

Aalto University
School of Science

HELSINKI INSTITUTE OF PHYSICS

Motivation

- Study the differences between ARM vs Intel in terms of energy consumption
- Commonly cited difference : Instruction Set (RISC vs CISC)
- Goal: Investigate whether x86-64 processors consume more energy because of their complex instruction set

Problem

How can we measure the energy consumption of instruction decoders?

Methodology

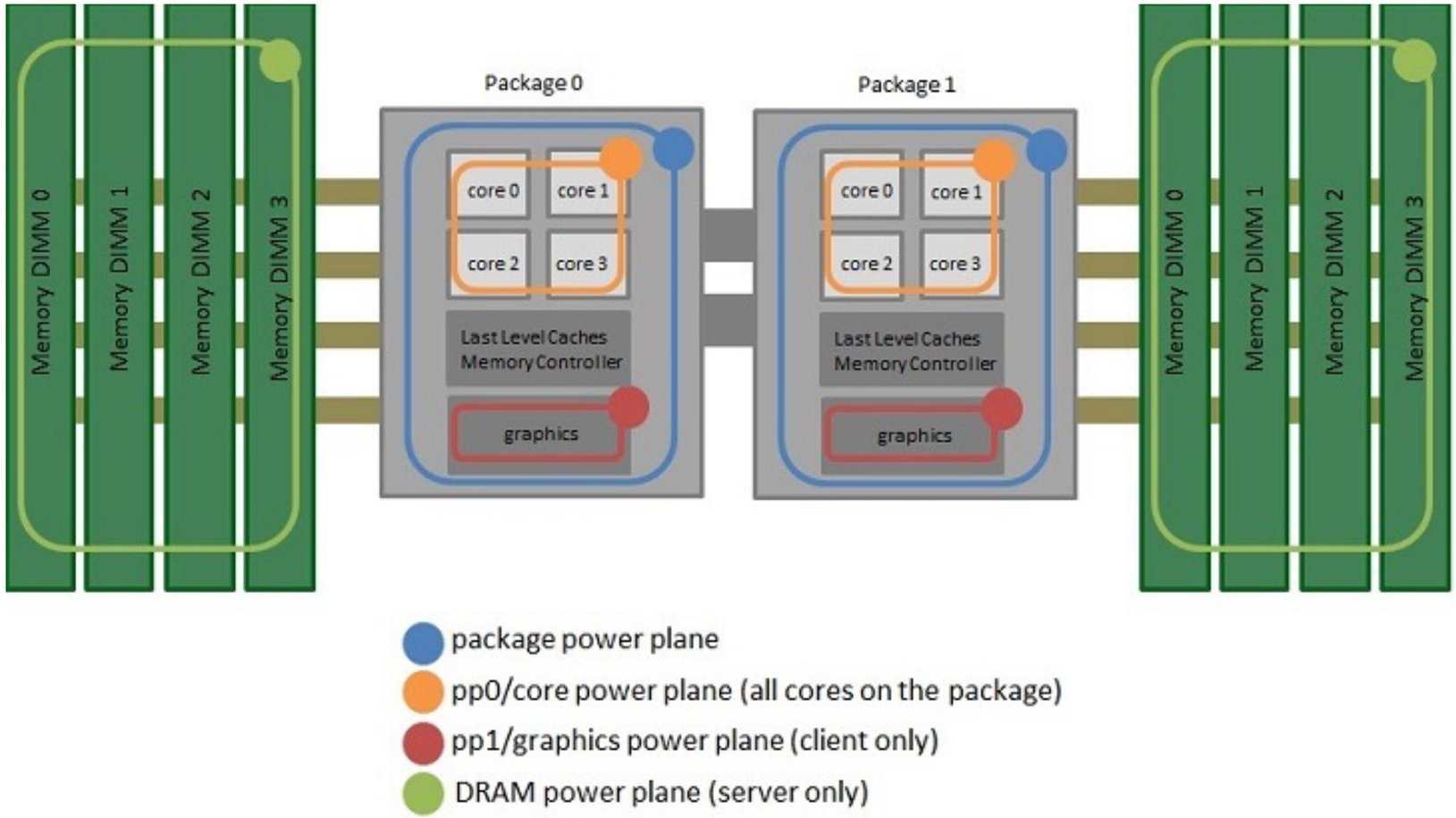
- Using micro-benchmarks that specifically trigger the instruction decoders by exceeding the capacity of the decoded instruction cache

- Leverage

  Intel’s RAPL (Running Average Power Limit)

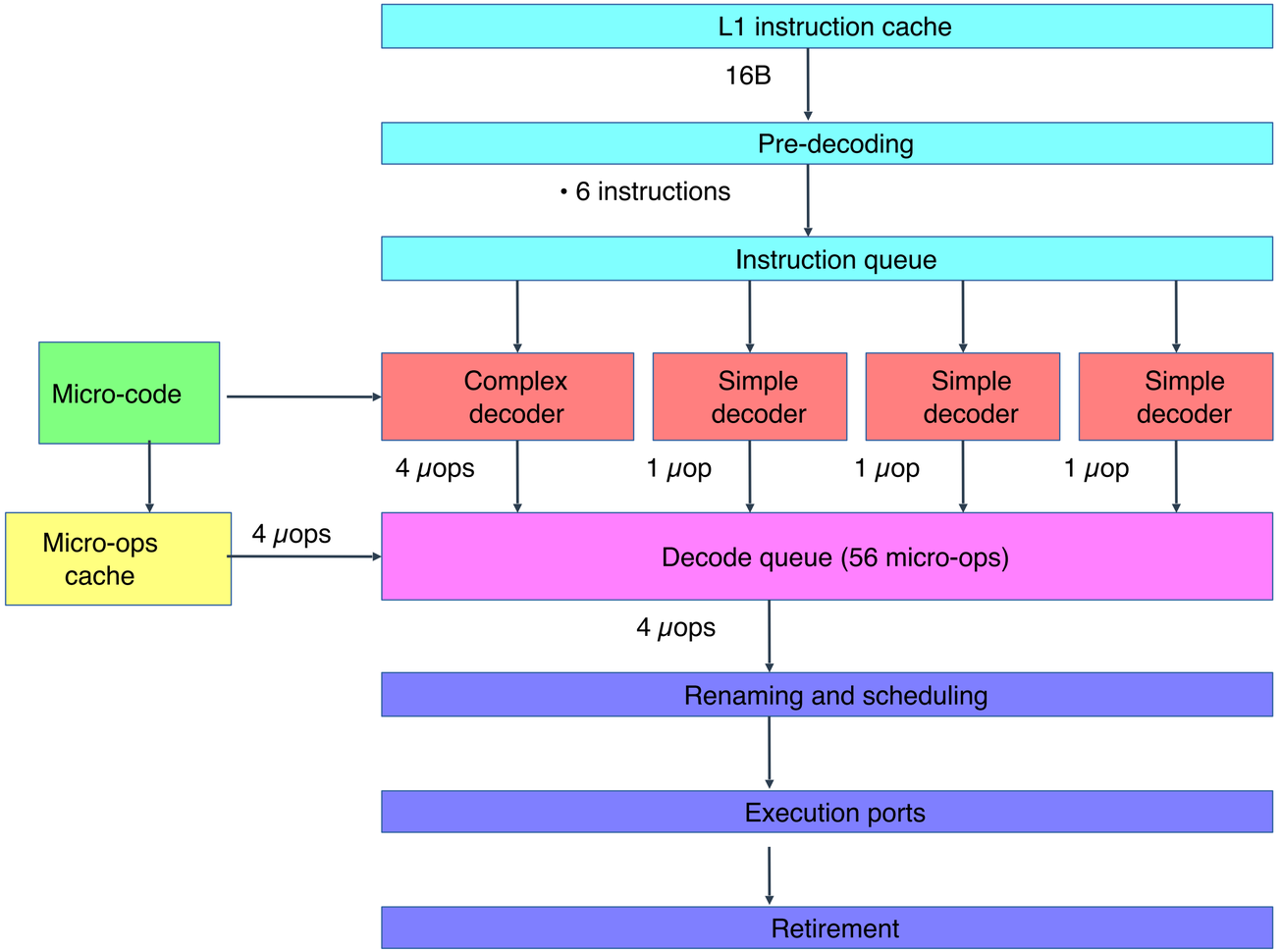
  Micro-op cache

Intel RAPL



https://software.intel.com/en-us/articles/intel-power-governor

Intel Haswell pipeline



Instruction decoder benchmarks

```
D += A[ j ] + B[ j ]   C[ j ] ;

D += (A[ j ] <<  3) *  (A[ j ] <<  4)  * ( (B[ j ] <<  2) *  5 + 1 ) ;
```

- Simple arithmetic operations
- Float and integer data types
- Arrays or registers
- Uses L1 or L2 caches
- Uses loop unrolling to increase the code size

Loop unrolling

Idea: Trigger the instruction decoders by using a big unroll count for a single loop

Original loop

```
for(int j=0;j<n;j++)
D += A[j] + B[j] * C[j]
```

Unrolled loop

```
for(int j=0;j<n;j+=256)
{
D += A[j] + B[j] * C[j]
D += A[j+1] + B[j+1] * C[j+1]
   …
D += A[j+ 255] + B[j+255] * C[j+255]
}
```

Advantages of Loop unrolling

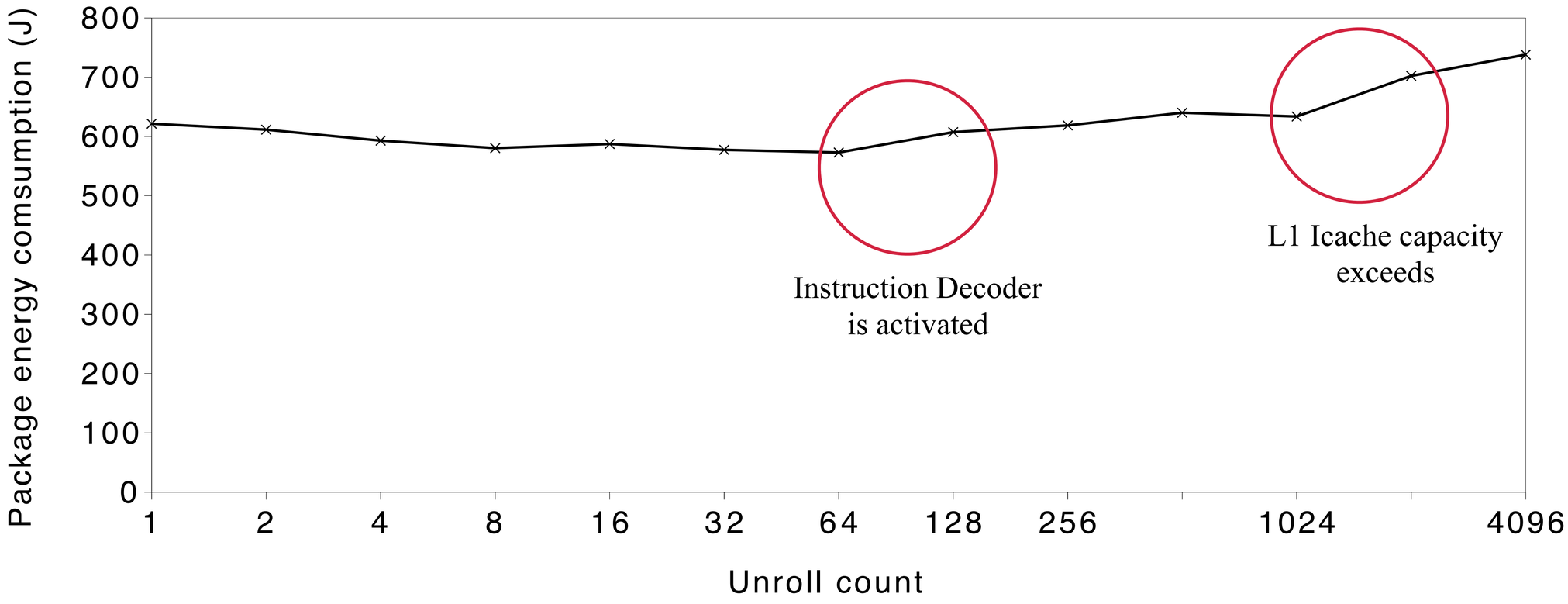
- Increases performance
- Opportunities to find instruction level parallelism

What we do: manually unroll the loop up to 2048 times

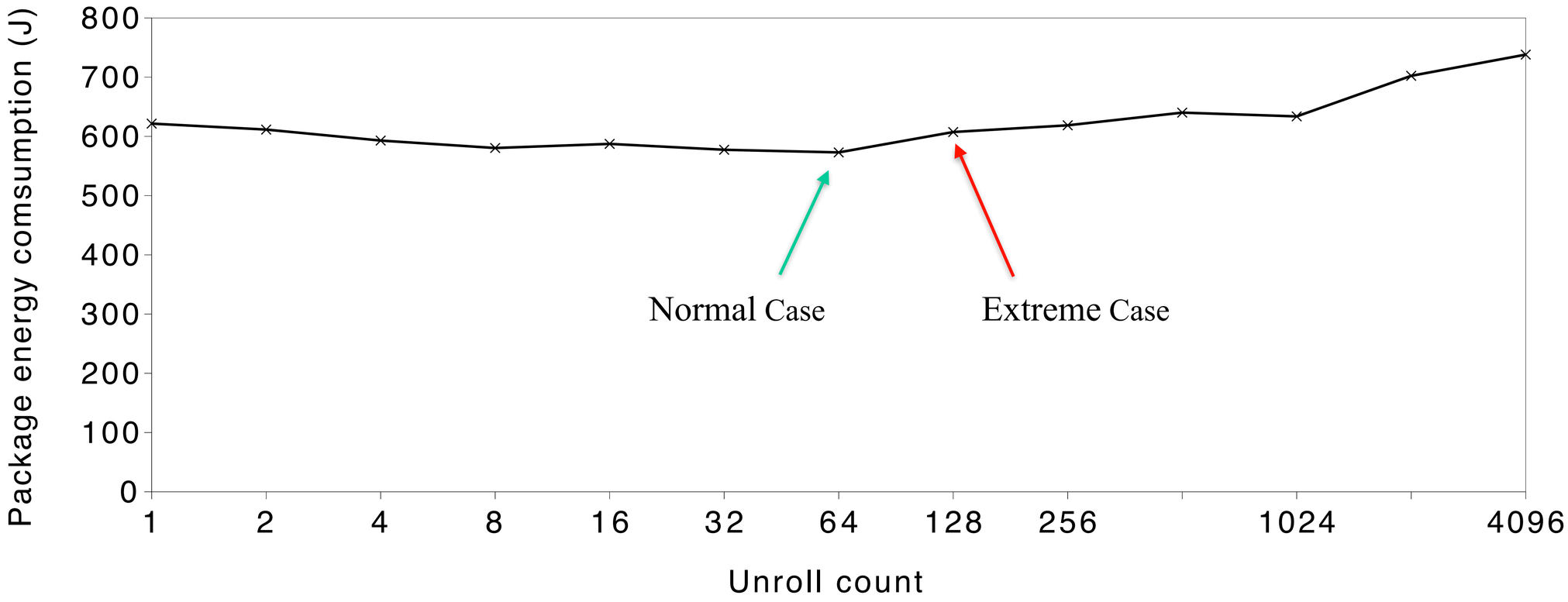
- Increases the code size
- Triggers instruction decoder

Idea: If the loop is still executed in the same time as before, the observed difference in power consumption can be attributed to instruction decoding pipeline

Evaluation - Power consumption vs. Code Size



Evaluation - Power consumption vs. Code Size



Power Modeling

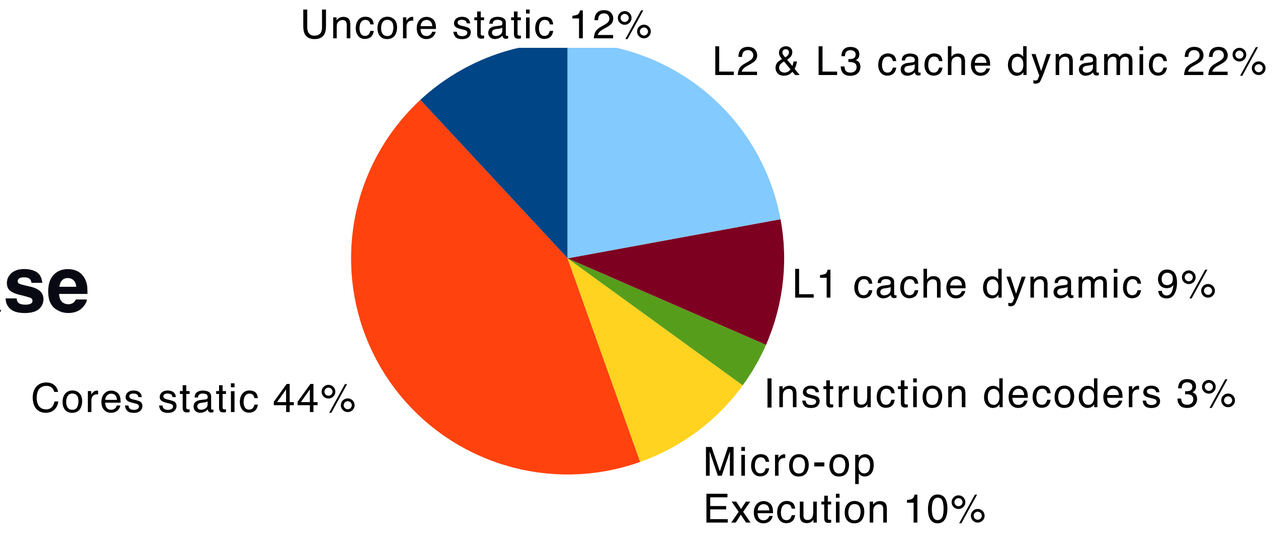
Event name:	Description:
CPU_CLK_UNHALTED.THREAD_P	The number of clock cycles for each core.
UOPS_ISSUED.ANY	The number of micro-ops issued to the execution units.
IDQ.MITE_UOPS	The number of micro-ops produced by the instruction decoders.
MEM_LOAD_UOPS_RETIRED.L1_HIT	The number of hits in the L1 data cache.
L2_RQSTS.REFERENCES	The number requests to the L2 cache (including L2 prefetchers).

$$P_{\text{package}} = 6.05 + \frac{\text{cycles}}{\text{second}} \times 1.63 \times 10^{-9} + \frac{\mu\text{ops issued}}{\text{second}} \times 2.15 \times 10^{-10} + \frac{\mu\text{ops decoded}}{\text{second}} \times 1.40 \times 10^{-10}$$

$$+ \frac{\text{L1 hits}}{\text{second}} \times 4.35 \times 10^{-10} + \frac{\text{L2 references}}{\text{second}} \times 4.05 \times 10^{-9}$$

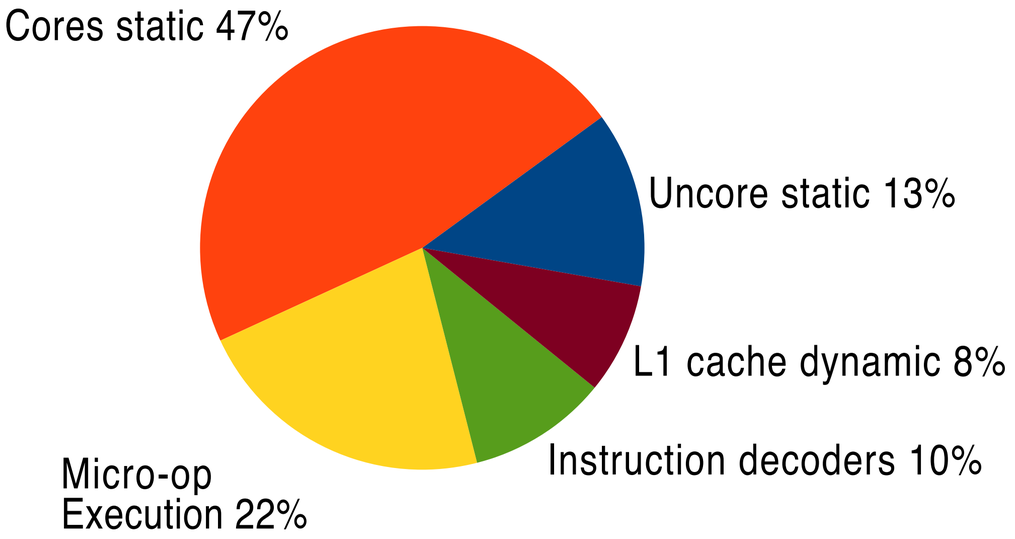
Evaluation- Power consumption breakdown

Normal case



Evaluation- Power consumption breakdown

Extreme case



Summary

The x86-64 instruction decoders consume relatively little power

Instruction decoder consumes 3% - 10% of the processor package power

The instruction set is not a significant drawback compared to ARM

We plan to port our benchmark to ARM platforms

Microbenchmark code

https://github.com/mhirki/idq-bench2

Green Big Data Project overview

Overall Research Focus

Energy efficiency analysis in High Performance Scientific Computing

- Port IgProf to 64-bit ARM
- Energy profiler module
- Techniques and tools for measuring energy efficiency
- Intel RAPL(Running Average Power Limit) evaluation

https://twiki.cern.ch/twiki/bin/view/Main/GreenBigData

Thank you!

Questions?