THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

PERFORMANCE, ENERGY, AND EVENT CHARACTERIZATION ON MODERN INTEL PROCESSORS

Aleksandar Milenkovic

(based on Ranjan Hebbar's work)

November 21, 2019

The Department of Electrical & Computer Engineering

LaCASA

Outline

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- Introduction
- SPEC CPU2017
- Test Environment & Profiling Tools
- Test Machine Setup
- SPEC CPU2017 Benchmarks Scalability Analysis and Characterization
- Architectural Comparison
- Conclusion

LaCASA

Introduction

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- Background & Motivation
  - Technology: Moore's Law and Dennard's Law do not hold anymore
  - Applications: Increasing sophistication and complexity
  - Markets: Shortening time-to-market
- Performance evaluation
  - Which machine is faster? How to optimize my application? What are application bottlenecks? How to build a faster machine?
  - Metrics: speed, throughput, performance, power, performance/power
  - Stakeholders: consumers, system performance engineers, application developers, CSE/CS researchers
- Need for standardized workloads: SPEC, EEMBC, Parsec, …

LaCASA

What Is This Research About?

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- *Know your machine*! What configuration would yield the best performance and save energy?
  - Explore impact of Compilers, Hardware Prefetching, Hyperthreading, Frequency Scaling on Performance and Energy Efficiency
- *Know your application*! What resources does your workload need to perform better? How do you make it faster?
  - Explores SPEC CPU2017 benchmark suites and its scope of application
  - Characterizes individual SPEC CPU2017 benchmarks
  - Analyzes performance bottlenecks of each benchmark using the Intel VTune Amplifier's Top-down Microarchitectural Analysis Method
  - Tests scalability of each benchmark
- *Machine Comparison*. How do different classes of machines handle similar workloads?
  - Evaluates performance of a set of desktop and server computers using SPEC CPU2017 metrics

LaCASA

SPEC CPU

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- Benchmarking is critical for evaluating current systems for bottlenecks and exploring architectural enhancements in future systems
- Standard Performance Evaluation Corporation (SPEC), was founded in 1988
- It is a non-profit organization that aims to “establish, maintain and endorse standardized benchmarks and tools to evaluate performance and energy efficiency” for computing systems
- There have been six iterations of the SPEC CPU benchmarks
- Each generation revises the suites to represent current and future workloads
- Though SPEC CPU is perceived as a CPU intensive benchmark suit it does emphasize memory and compiler efficiency as well

LaCASA

SPEC CPU2017

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- What is SPEC CPU2017?
  - The SPEC CPU2017 benchmark suites contains SPEC's latest, industry-standardized, CPU intensive suites for measuring and comparing compute intensive performance, stressing a system's processor, memory subsystem, and compiler
- Why would I want to use it?
  - To evaluate performance of modern computers using "*speed*" (defined as how fast the processor can execute a certain task) and "*throughput*" (defined as the amount of work that is completed in a given amount of time) metrics for compute-intensive workloads
- What will I learn from the results?
  - How the system under test compares with the reference machine
  - Possible bottlenecks affecting performance

LaCASA

SPEC CPU2017 Benchmarks

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

SPECrate 2017 Floating Point	SPECspeed 2017 Floating Point	Language	Application Area
503.bwaves_r	603.bwaves_s	Fortran	Explosion modeling
507.cactuBSSN_r	607.cactuBSSN_s	C++, C, Fortran	Physics: relativity
508.namd_r		C++	Molecular dynamics
510.parest_r		C++	Biomedical imaging: optical tomography
511.povray_r		C++, C	Ray tracing
519.lbm_r	619.lbm_s	C	Fluid dynamics
521.wrf_r	621.wrf_s	Fortran, C	Weather forecasting
526.blender_r		C++, C	3D rendering and animation
527.cam4_r	627.cam4_s	Fortran, C	Atmosphere modeling
	628.pop2_s	Fortran, C	Wide-scale ocean modeling (climate level)
538.imagick_r	638.imagick_s	C	Image manipulation
544.nab_r	644.nab_s	C	Molecular dynamics
549.fotonik3d_r	649.fotonik3d_s	Fortran	Computational Electromagnetics
554.roms_r	654.roms_s	Fortran	Regional ocean modeling

SPECrate 2017 Integer	SPECspeed 2017 Integer	Language	Application Area
500.perlbench_r	600.perlbench_s	C	Perl interpreter
502.gcc_r	602.gcc_s	C	GNU C compiler
505.mcf_r	605.mcf_s	C	Route planning
520.omnetpp_r	620.omnetpp_s	C++	Discrete Event simulation - computer network
523.xalancbmk_r	623.xalancbmk_s	C++	XML to HTML conversion via XSLT
525.x264_r	625.x264_s	C	Video compression
531.deepsjeng_r	631.deepsjeng_s	C++	Artificial Intelligence: alpha-beta tree search (Chess)
541.leela_r	641.leela_s	C++	Artificial Intelligence: Monte Carlo tree search (Go)
548.exchange2_r	648.exchange2_s	Fortran	Artificial Intelligence: recursive solution generator (Sudoku)
557.xz_r	657.xz_s	C	General data compression

LaCASA

SPEC Metrics: What is in a Number?

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- Reference machine (RM): Sun Microsystems server, the Sun Fire V490 with 2100 MHz UltraSPARC-IV+ chips
- Speed ratio (1 copy of benchmark is run, tester chooses the number of OpenMP threads)
- $SR(SBi) = \frac{ET(SBi,RM)}{ET(SBi,SUT)}$
- SPECspeed2017_fp_base, SPECspeed2017_int_base: Geometric mean of individual speed ratios
- Throughput ratio (tester chooses N_C copies to run; OpenMP is disabled)
- $TR(RBi) = \frac{N_C \cdot ET(RBi,RM)}{ET(SBi,SUT)}$
- SPECrate2017_fp_base, SPECrate2017_int_base: Geometric mean of individual speed ratios

LaCASA

More on SPEC CPU2017

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- Landing page: https://www.spec.org/cpu2017/
- Overview: https://www.spec.org/cpu2017/Docs/overview.html
- Submitted results: https://www.spec.org/cpu2017/results/

LaCASA

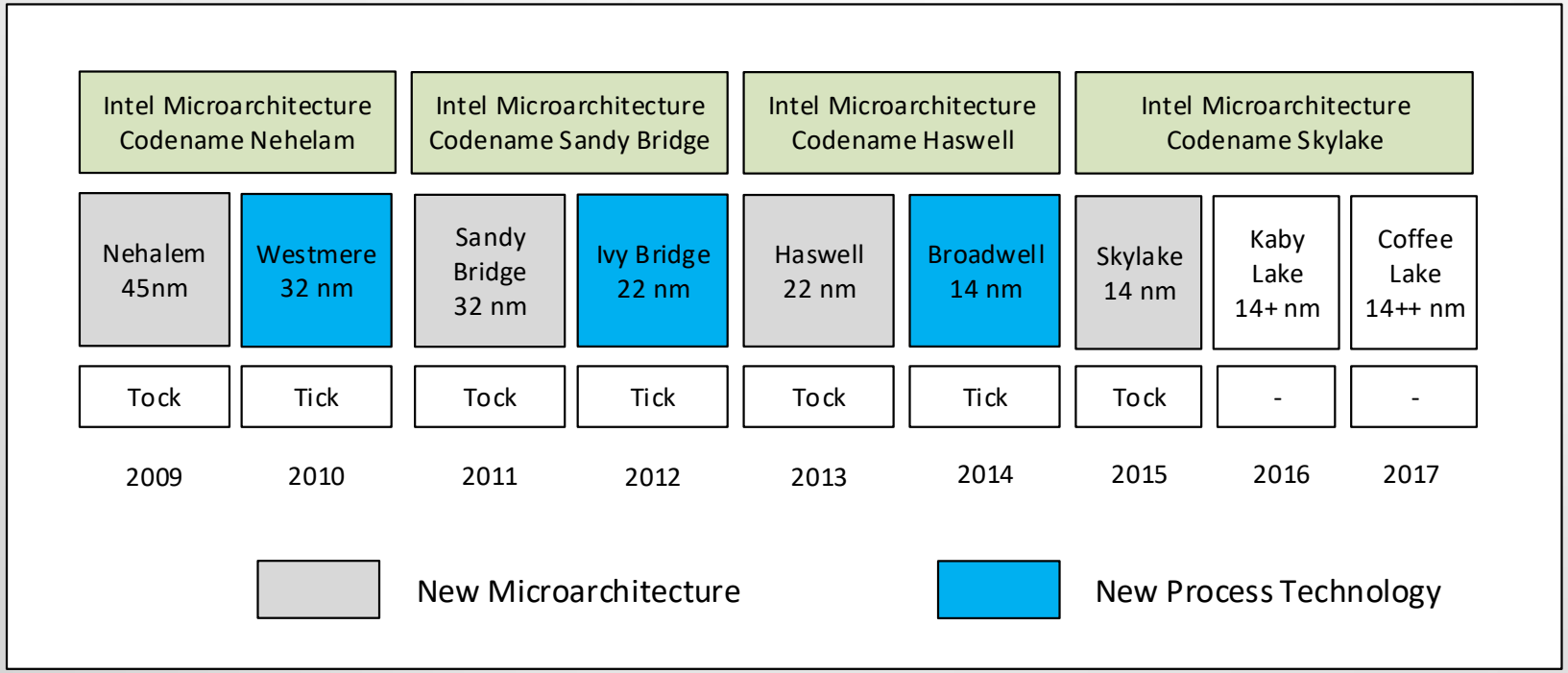
Test Environment & Profiling Tools

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- The organization of this subsection is as follows
  - Microarchitectural overview
  - Intel turbo-boost technology
  - Systems under test
  - Compilers used for the study
  - Tools and application used in this study
  - Top-down Microarchitectural Analysis Method

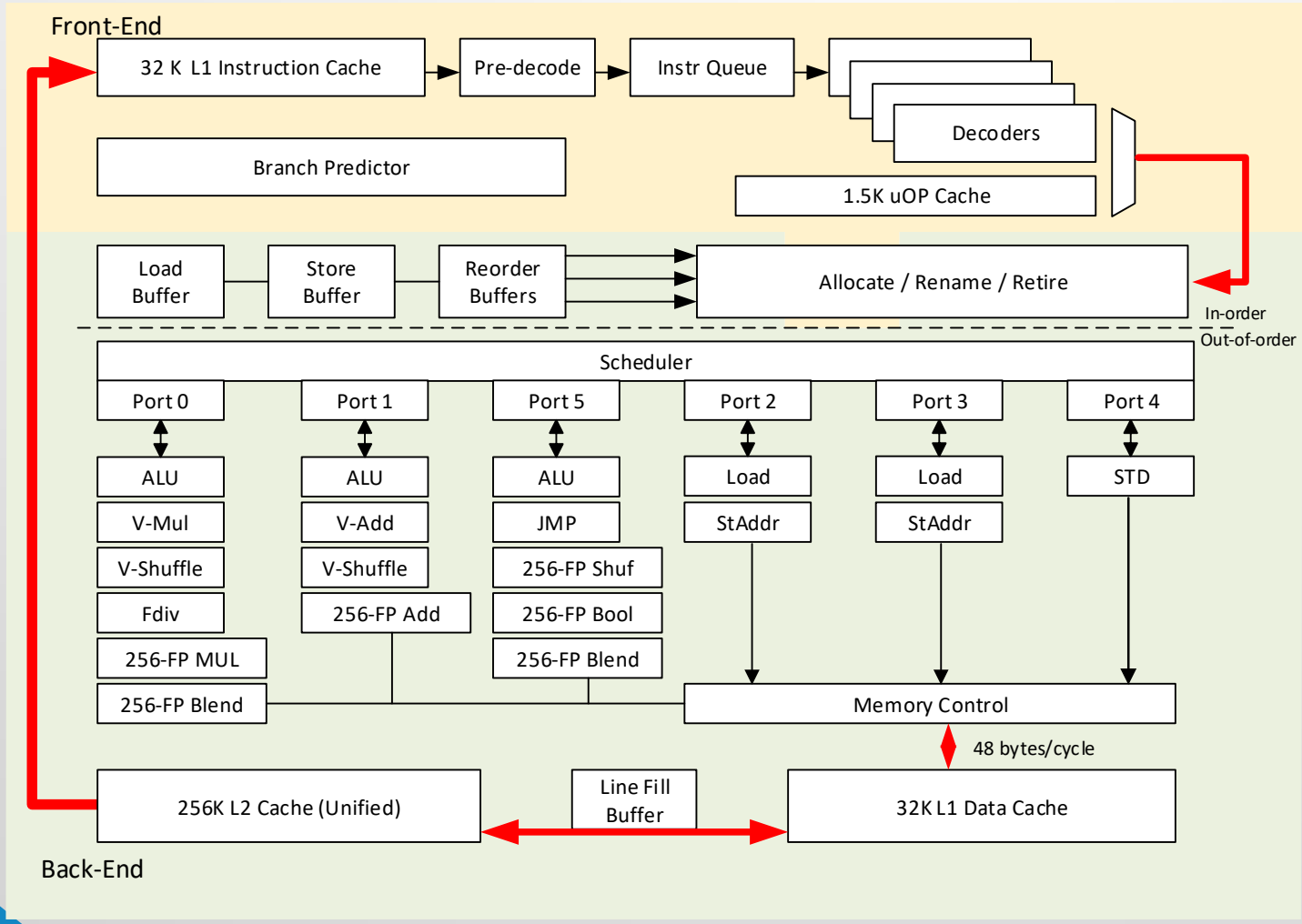
Intel Processors: Tick-Tock

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

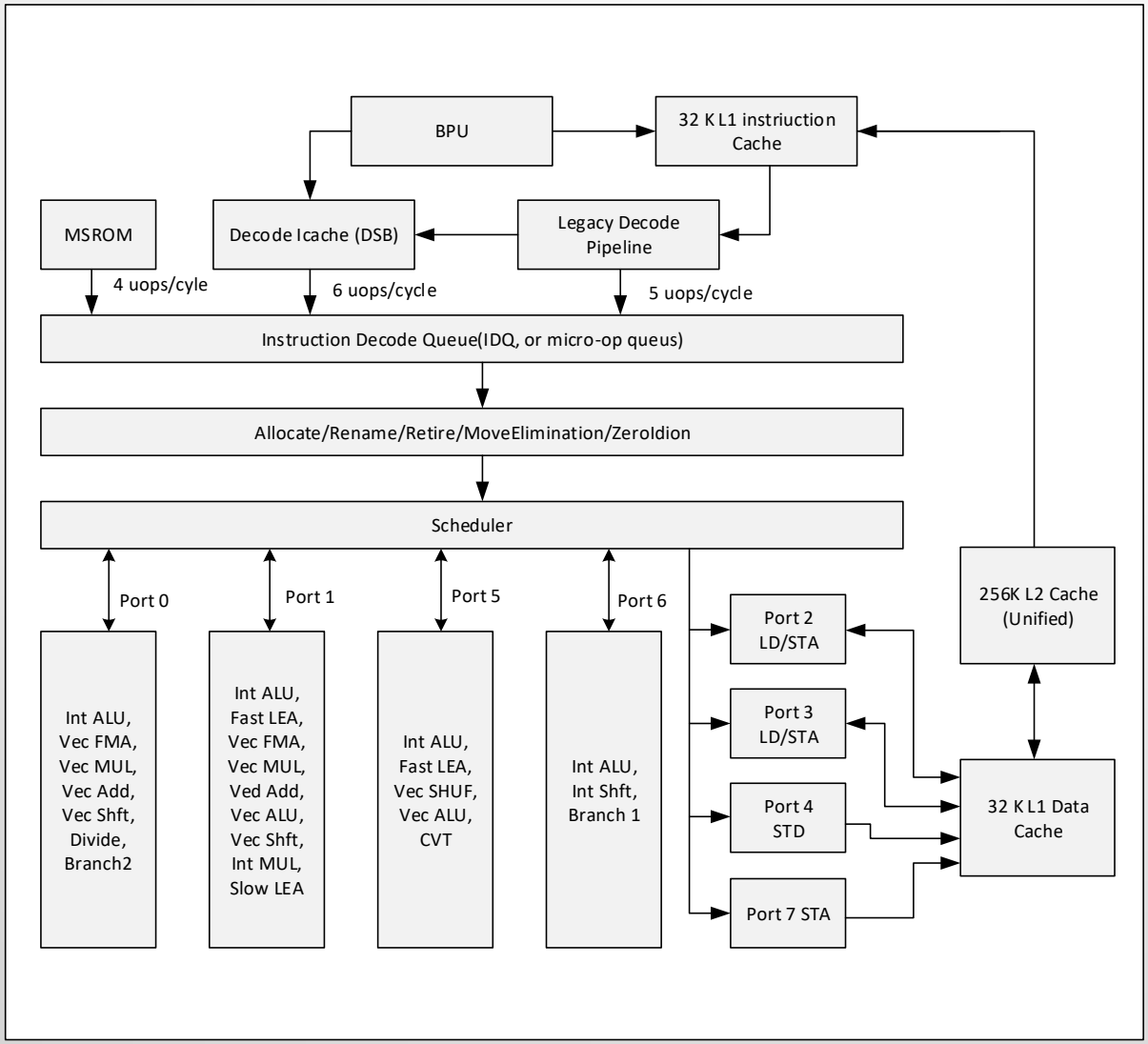


LaCASA

Intel Microarchitecture: Sandy Bridge

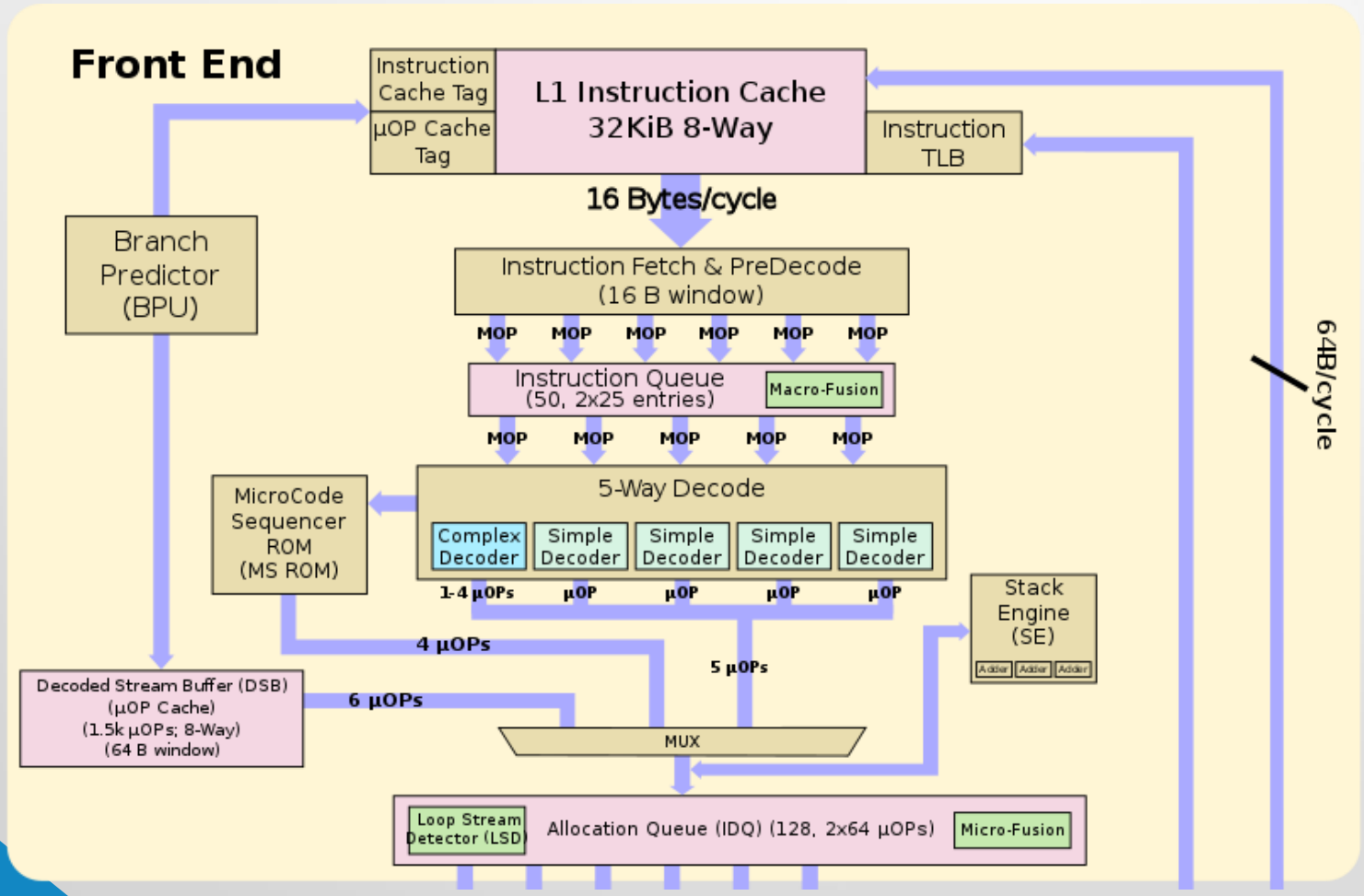


Intel Microarchitecture: Skylake



Coffee Lake (Front-End)

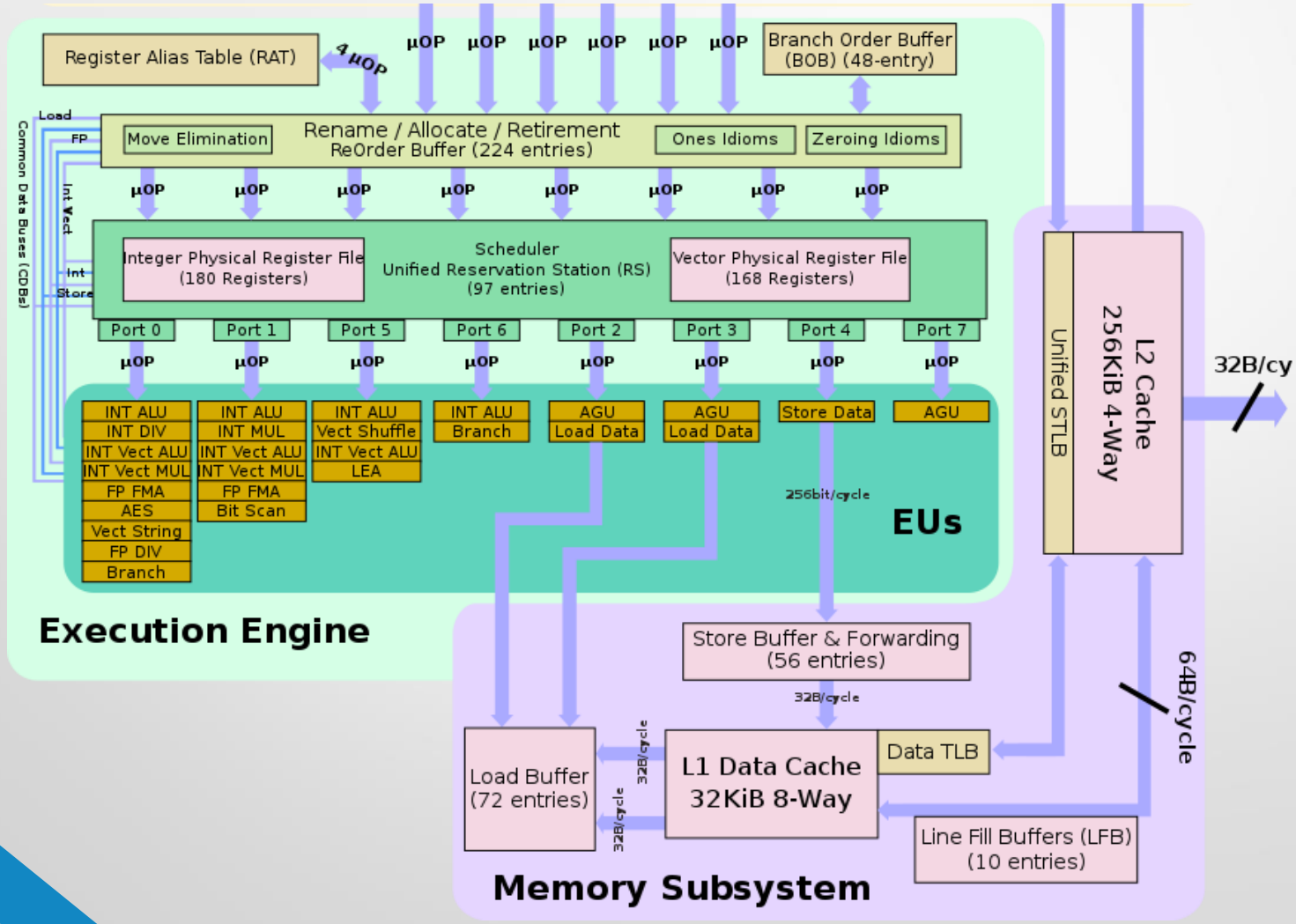
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

Coffee Lake (Back-End)

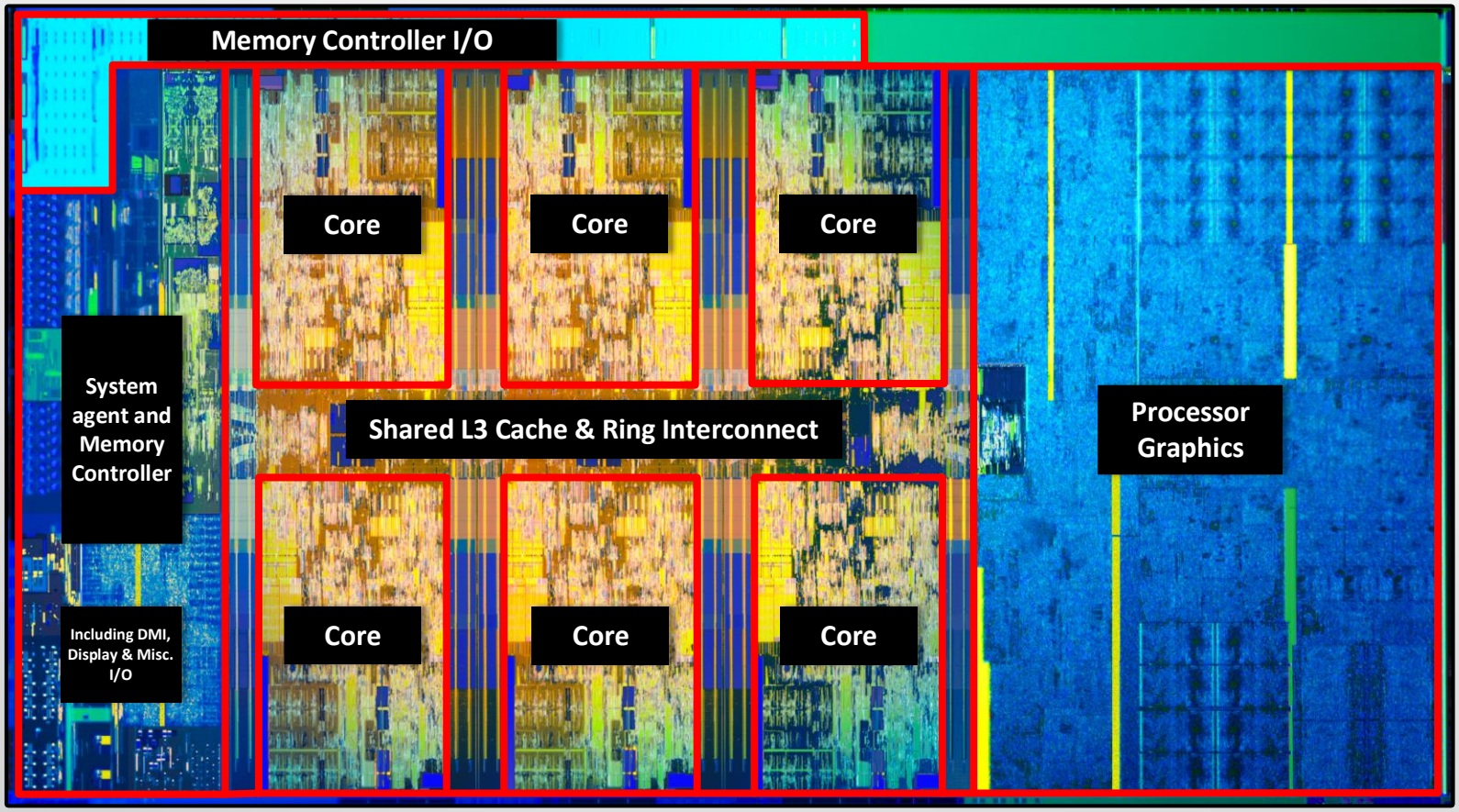
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

Annotated Die Map of Core i7-8700K

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

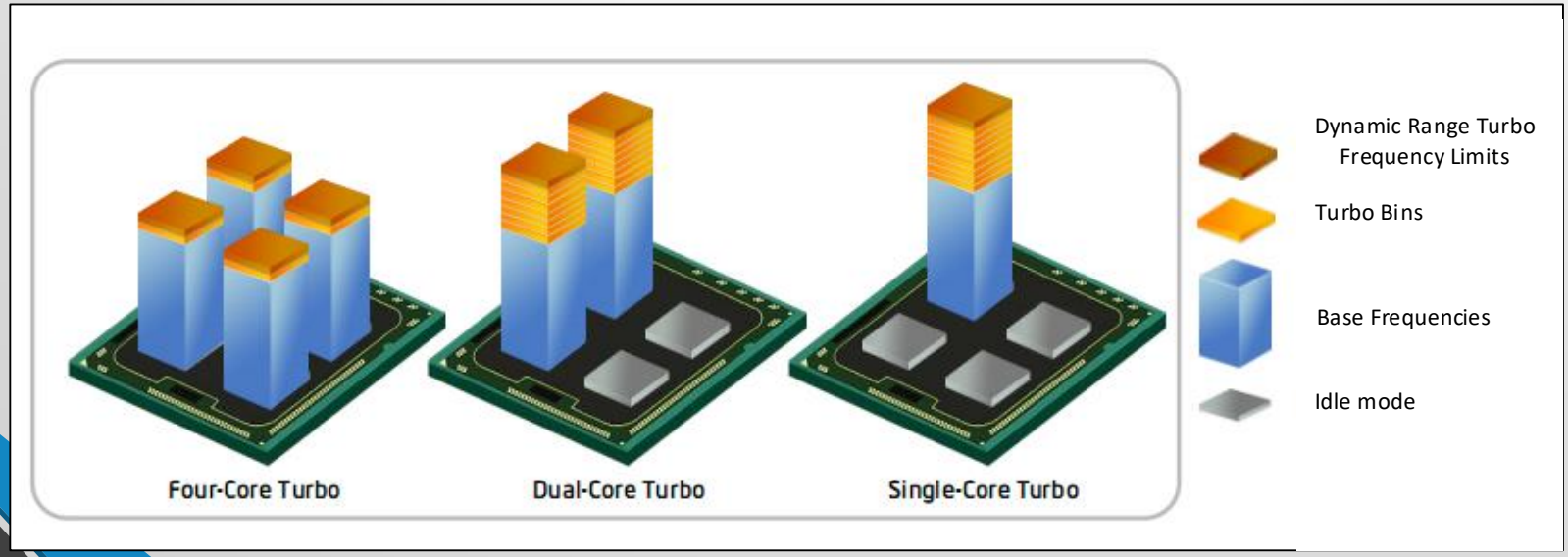


LaCASA

Intel Turbo-Boost Technology 2.0

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- Turbo mode accelerates processor and onboard graphics performance for peak loads
- With lower leakage current the core frequency could be increased resulting in a larger frequency allocation for turbo mode
- A high frequency operation is costly in terms of power and increases thermal dissipation. Hence turbo mode is used only when required



LaCASA

Systems Under Test

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

Sys. Code name	*mtsano*	*clingmansdome*	*vidrak*	*mtleconte*
Processor	Core i7-4770	Core i7-8700K	Xeon E3-1240 v2	Xeon E5-2643 v3
Lithography	22nm	14nm	22nm	22nm
Generation	4th Generation	8th Generation	3th Generation	4th Generation
Intel Codename	Haswell	Coffee-Lake	Ivy-Bridge	Haswell
Year of release	Q2-2013	Q4-2017	Q2-2012	Q3-2014
Physical Cores	4	6	4	6*2
Threads / Core	2	2	2	2
Logical Cores	8	12	8	12*2
CPU Max Freq.	3.9 GHz	4.70 GHz	3.8 GHz	3.7 GHz
CPU Avg. Freq.	3.4 GHz	3.7 GHz	3.4 GHz	3.4 GHz
CPU Min Freq.	0.8 GHz	0.8 GHz	1.6 GHz	1.2 GHz
L1d cache	32K*4	32K*6	32K*4	32K*6*2
L1i cache	32K*4	32K*6	32K*4	32K*6*2
L2 Cache	256K*4	256K*6	256K*4	256K*6*2
L3 Cache (LLC)	8192K	12288K	8192K	20480K
RAM	16 GB	32 GB	16GB	64 GB
RAM Freq.	1600 MHz	2400 MHz	1600 MHz	2400 MHz
TDP (watts)	84 W	95 W	69 W	135 W
Operating System	Ubuntu 16.04.4 LTS	Ubuntu 16.04.4 LTS	Ubuntu 16.04.4 LTS	Ubuntu 16.04.4 LTS
OS Codename	Xenial	Xenial	Xenial	Xenial
Kernel	4.13.0-37-generic	4.4.0-116-generic	4.4.0-116-generic	4.4.0-116-generic
icc version	18.0.1	18.0.1	18.0.1	18.0.1
Perf version	4.13.13	4.4.98	4.4.98	4.4.98

LaCASA

Compilers Used

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- *Intel Parallel Studio XE-18 (IPS)*
- Developed by the Intel Corporation to facilitate code development in C, C++, and Fortran for parallel computing
- IPS is a hardware aware compiler and it is optimized for x86 instruction set architecture (ISA)
- –O3 optimization level, IPS enables auto-parallelization and vectorization

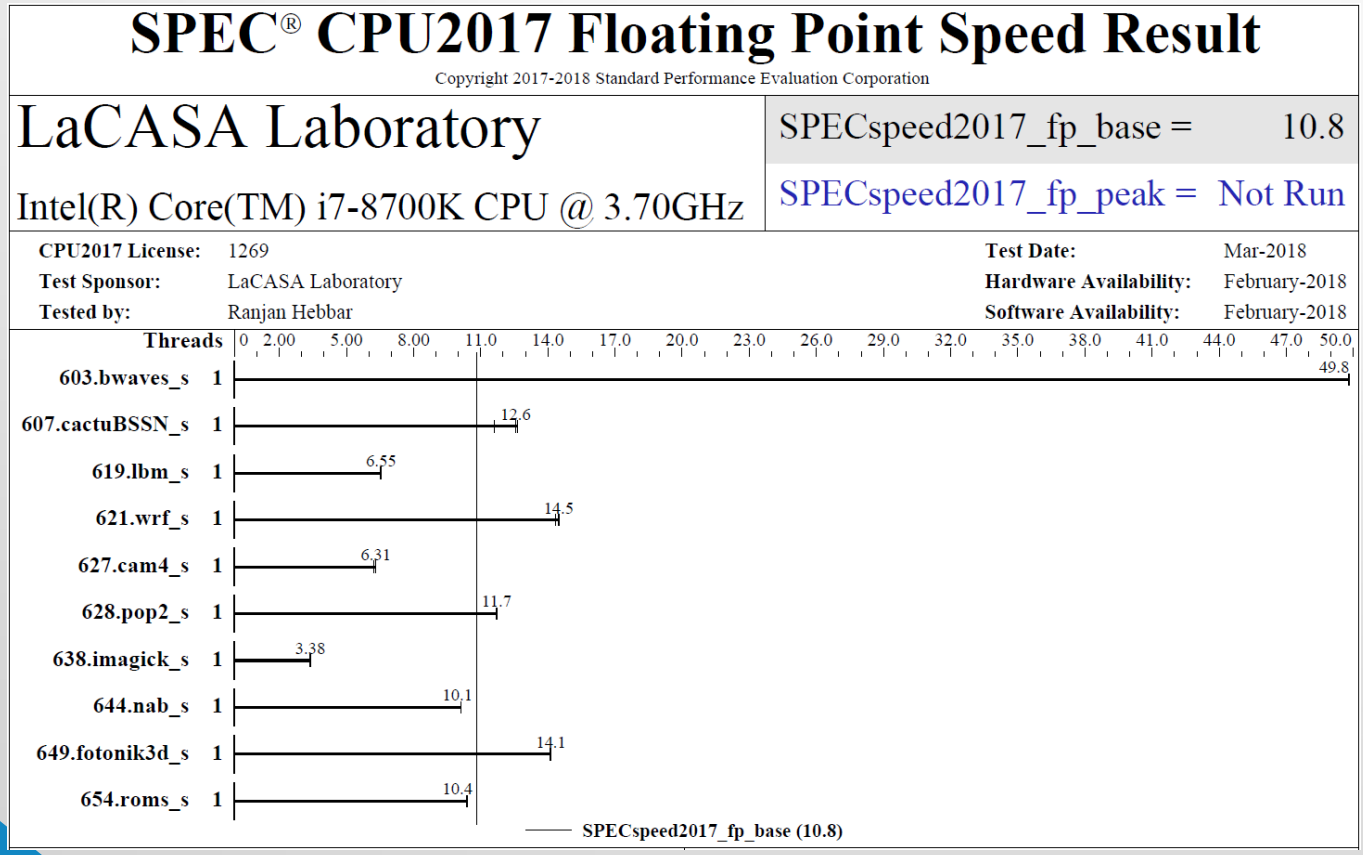
- *GNU Compiler Collection*
- The GNU compiler collection is an open source compiler from the Free Software Foundation Inc. which was originally written to be a compiler for the GNU operating system
- It is widely used across various target architectures and is distributed along with the Linux operating systems
- GNU compiler collection includes front ends for many programming languages and libraries from these languages

LaCASA

Tools and Applications: runcpu

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- SPEC utility that runs all the input sets for all the benchmarks three times and compiles a single SPEC number for analysis



LaCASA

Tools and Applications: perf

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

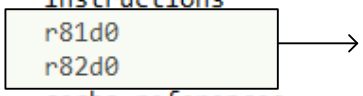
- Simple command line interface built into the OS allowing access to the Performance Monitoring Unit (PMU)
- Provides a list of measurable micro-architectural events
- Supports inheritance by default and has per thread/per process mode as well as system wide event collection

```
# started on Mon Nov 18 10:14:40 2019


Performance counter stats for '/home/rr0062/cpu2017/Rate/float/544_nab_1':

     895172954029      cycles
    1383150124323      instructions              #    1.55  insn per cycle
     482163094943      r81d0
     131835963416      r82d0
      10365495259      cache-references
       2839125043      cache-misses              #   27.390 % of all cache refs
     149994944477      branches
       6090376665      branch-misses             #    4.06% of all branches
            38997      page-faults
                7      cpu-migrations
             7999      context-switches

    242.200514109 seconds time elapsed
```



LaCASA

Tools and Applications: likwid

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- Has a set of tools with specific purposes to measure performance/energy groups
- Has added functionality over perf such as binding and measuring events on a per core basis

```
--------------------------------------------------------------------------------
CPU name:       Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz
CPU type:       Intel Kabylake processor
CPU clock:      3.70 GHz
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Runtime: 1356.99 s
Measure for socket 0 on CPU 0
Domain PKG:
Energy consumed: 23661.9 Joules
Power consumed: 17.4371 Watt
Domain PP0:
Energy consumed: 14451 Joules
Power consumed: 10.6493 Watt
Domain PP1:
Energy consumed: 0 Joules
Power consumed: 0 Watt
Domain DRAM:
Energy consumed: 0 Joules
Power consumed: 0 Watt
--------------------------------------------------------------------------------
```


LaCASA

Tools and Applications: VTune Amplifier

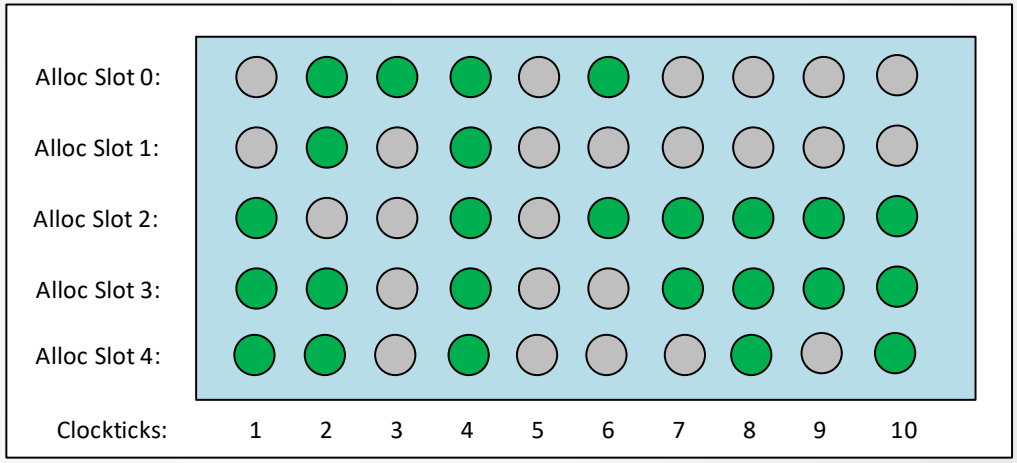
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

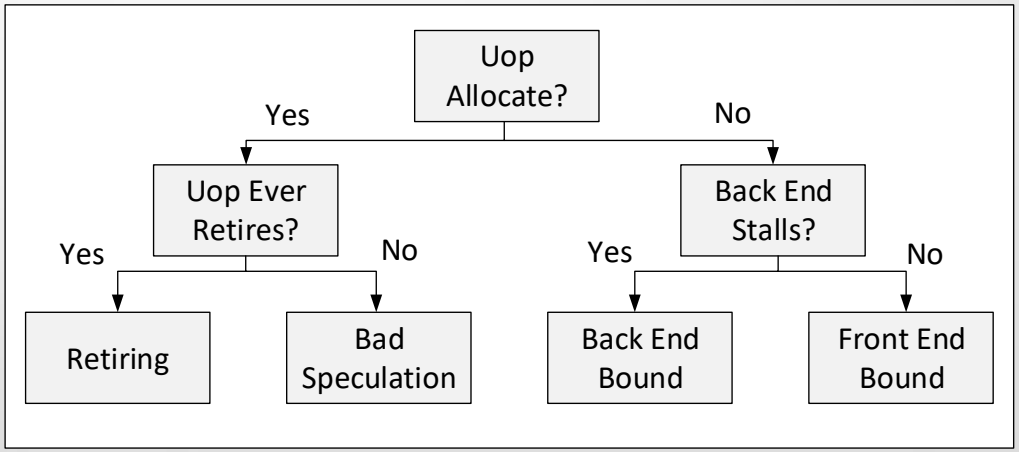
- Performance analysis tool that relies on the underlying hardware counters to get run-time parameters of the application under test
- Advanced Hotspot Analysis: Identify performance-critical code sections in the given application
- HPC Performance Characterization: Identify how effectively a compute-intensive application uses CPU, memory, and floating-point operation hardware resources
- Memory Access Analysis: Identify memory-related issues, like NUMA problems and bandwidth-limited accesses
- *General Exploration Analysis*: Understand how efficiently the code passes through the core pipeline

Top-down Microarchitectural Analysis Method

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- Top-down Microarchitectural Analysis by Ahmad Yasin from Intel
- Gives a pipeline view of the application flow
- Consider 5-wide pipeline for 10 cycles
- 50% of the slots are eventually retired
- The pipeline efficiency is 50%
- Each stalled pipeline slot can be attributed to one of three causes
  - Front-end Bound
  - Back-end Bound
  - Bad Speculation
- Ideally high retiring is preferred

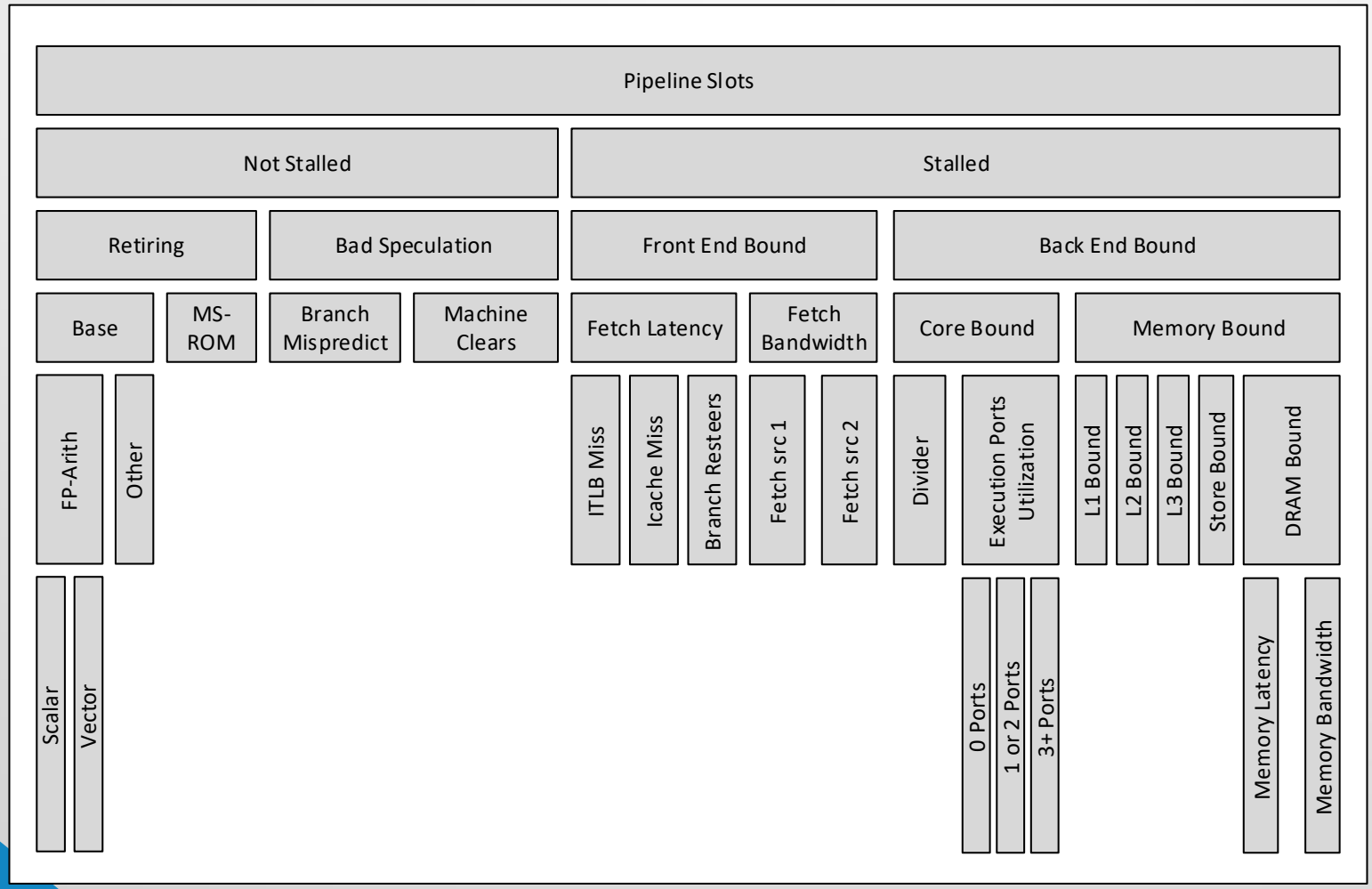




LaCASA

Top-down Microarchitectural Analysis Method

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



Scalability Study Metrics

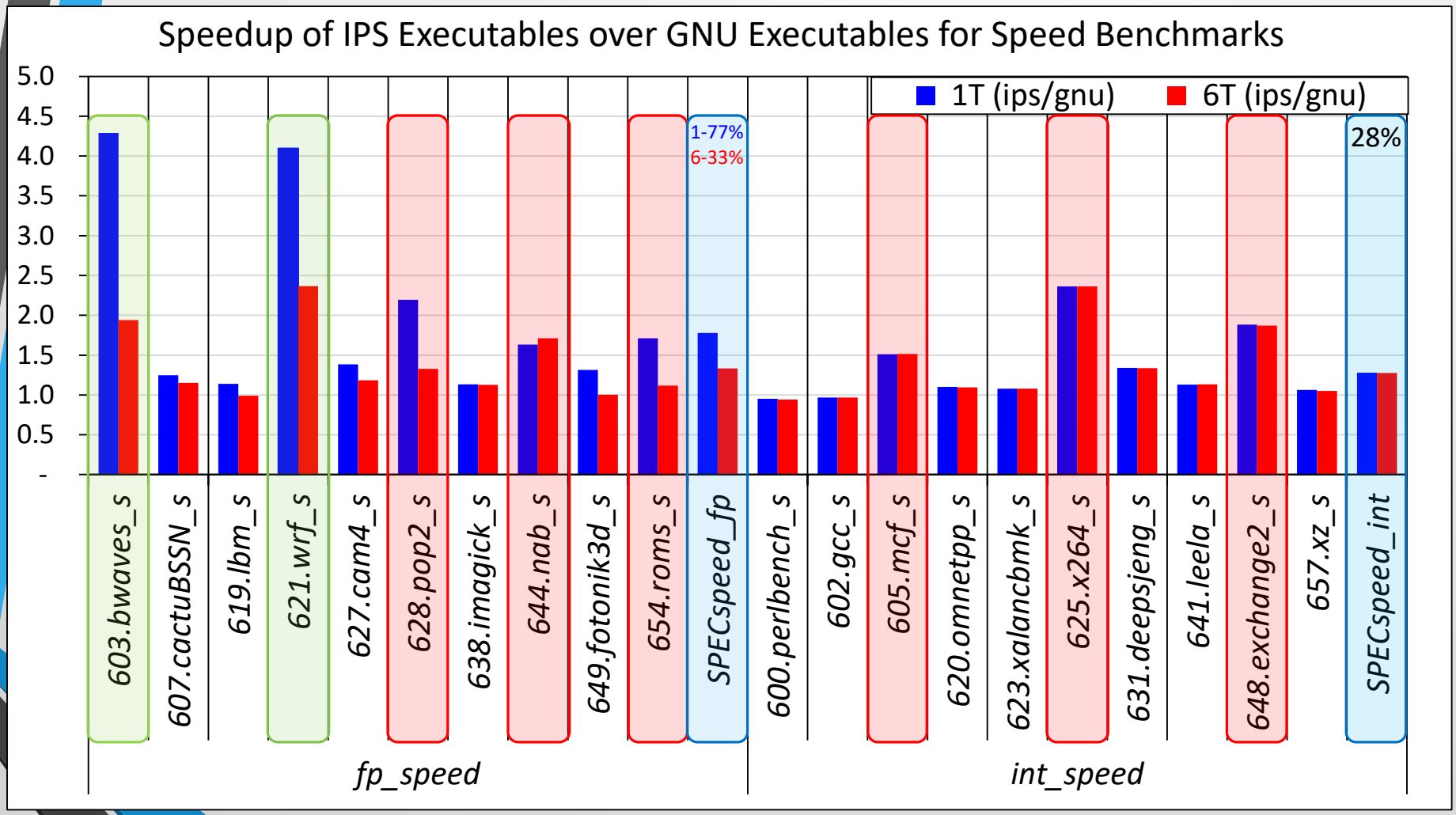
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

$Speedup(SBi, N_T) = \frac{Time(SBi, 1)}{Time(SBi, N_T)}$	**S**
$Speedup(RBi, N_C) = \frac{N_C * Time(RBi, 1)}{Time(RBi, N_C)}$	**S**
$Performance * Energy\ Efficiency(SBi, N_T) = \frac{1}{Time(SBi, N_T) \times Energy(SBi, N_T)}$	**PE**
$Performance * Energy\ Efficiency\ (RBi, N_C) = \frac{1}{Time(RBi, N_C) \times Energy(RBi, N_C)}$	**PE**
$Performace * Energy.Improvement(SBi, N_T) = \frac{PE(SBi, N_T)}{PE(SBi, 1)}$	**PE.I**
$Performace * Energy.Improvement(RBi, N_C) = \frac{{N_C}^2 * PE(RBi, N_C)}{PE(RBi, 1)}$	**PE.I**

LaCASA

Compiler Comparison For Speed Benchmarks

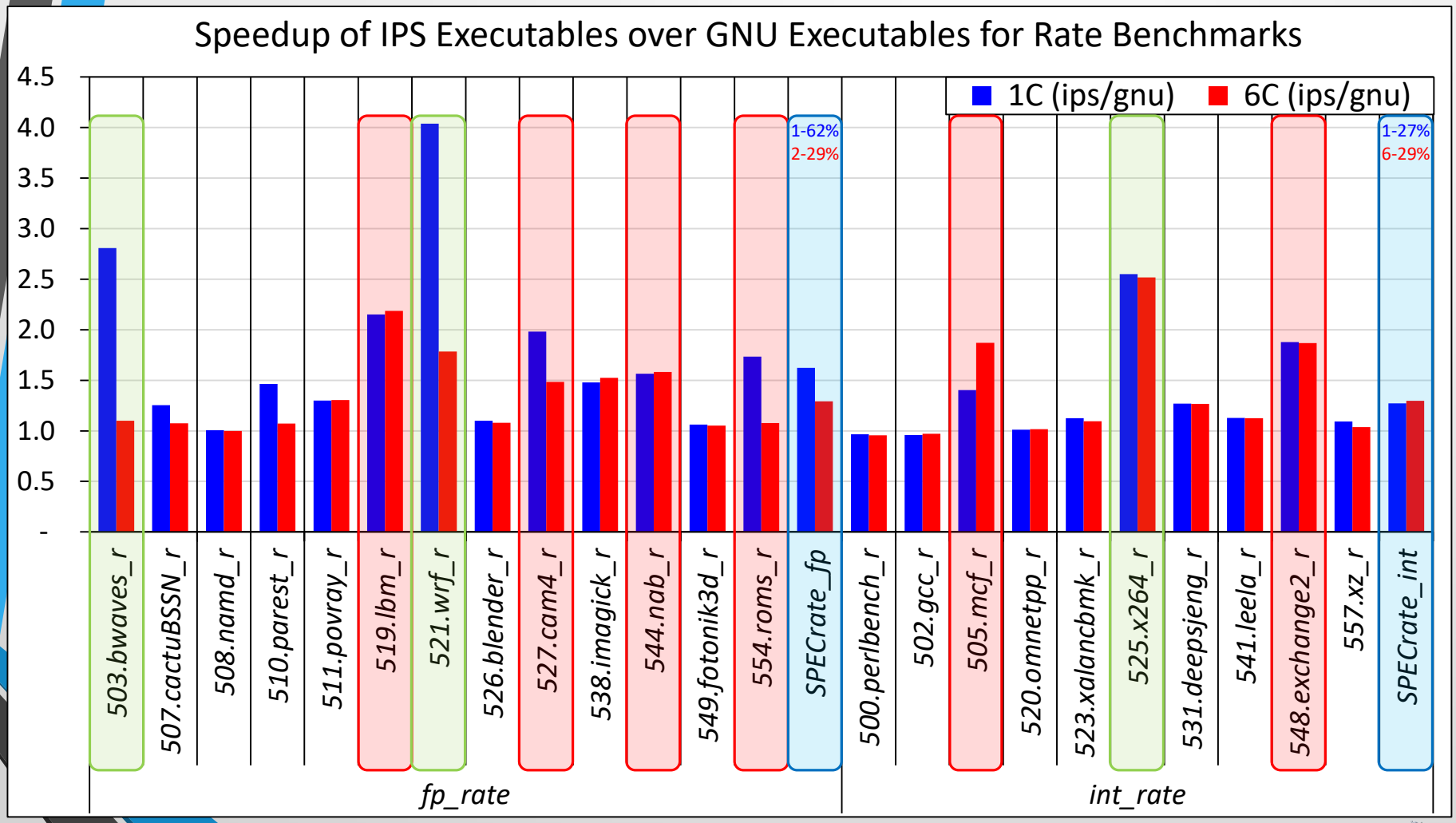
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

Compiler Comparison For Rate Benchmarks

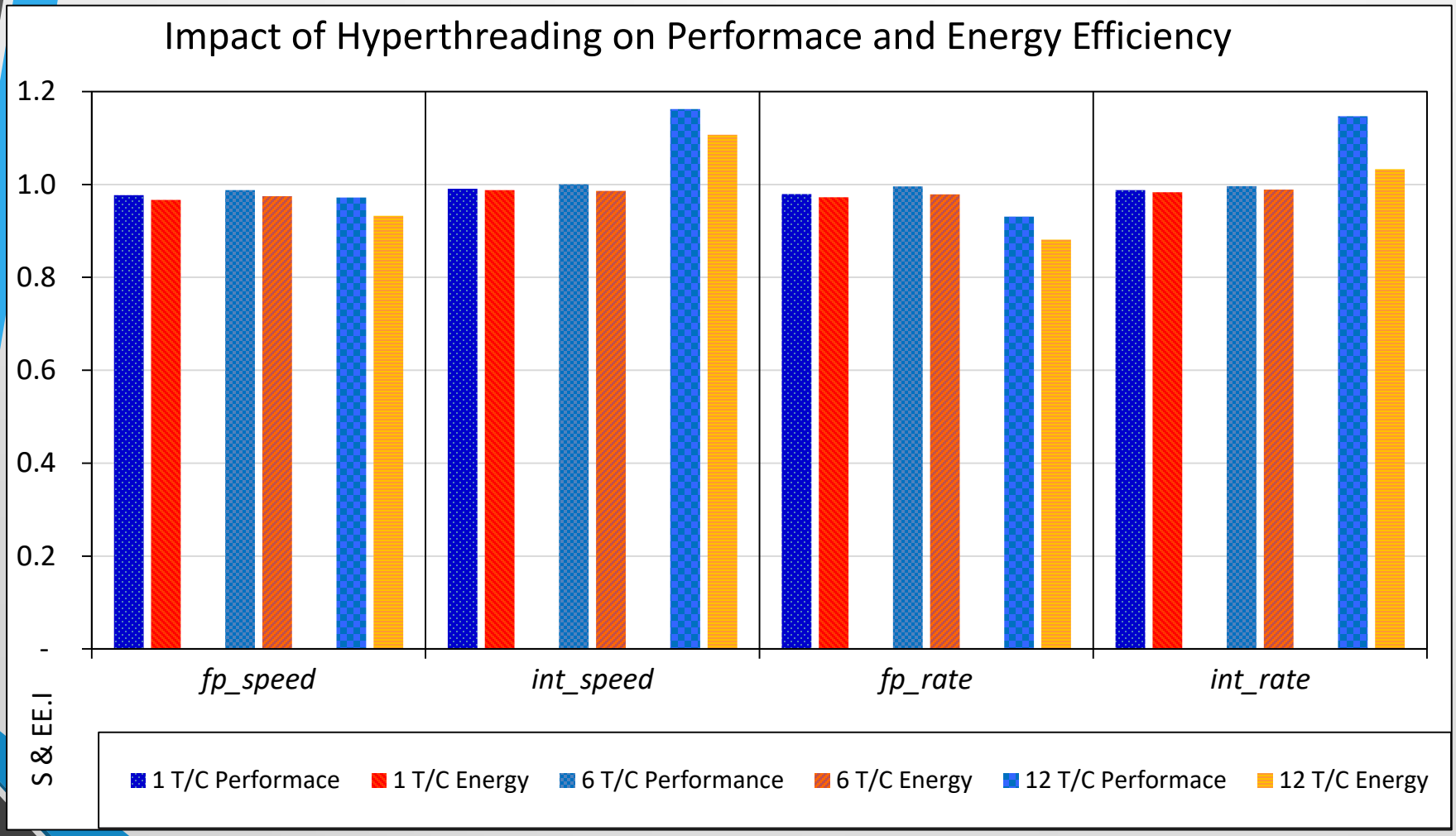
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

Test Setup: Why to turn off Hyperthreading

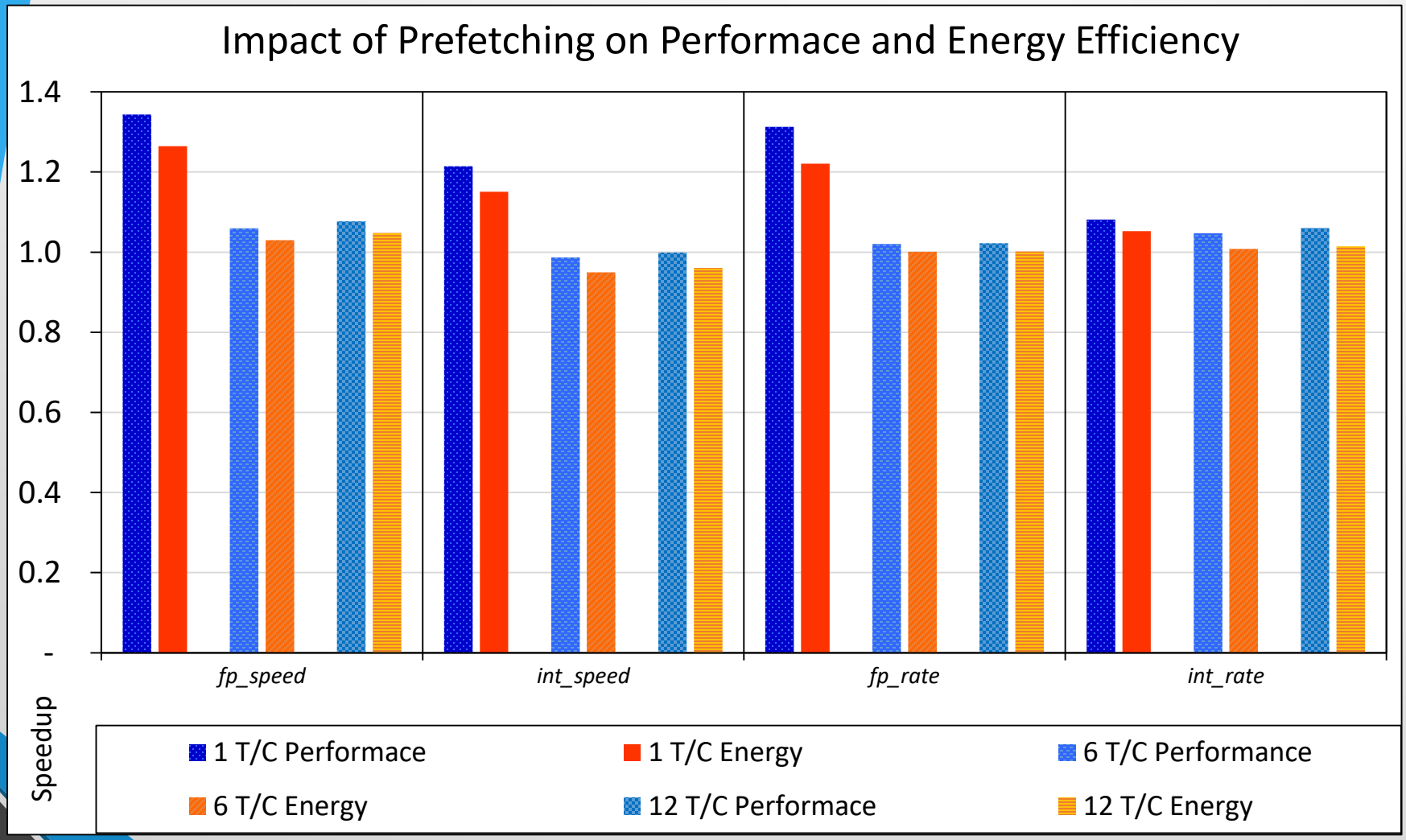
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

Test Setup: Why to turn on Prefetching

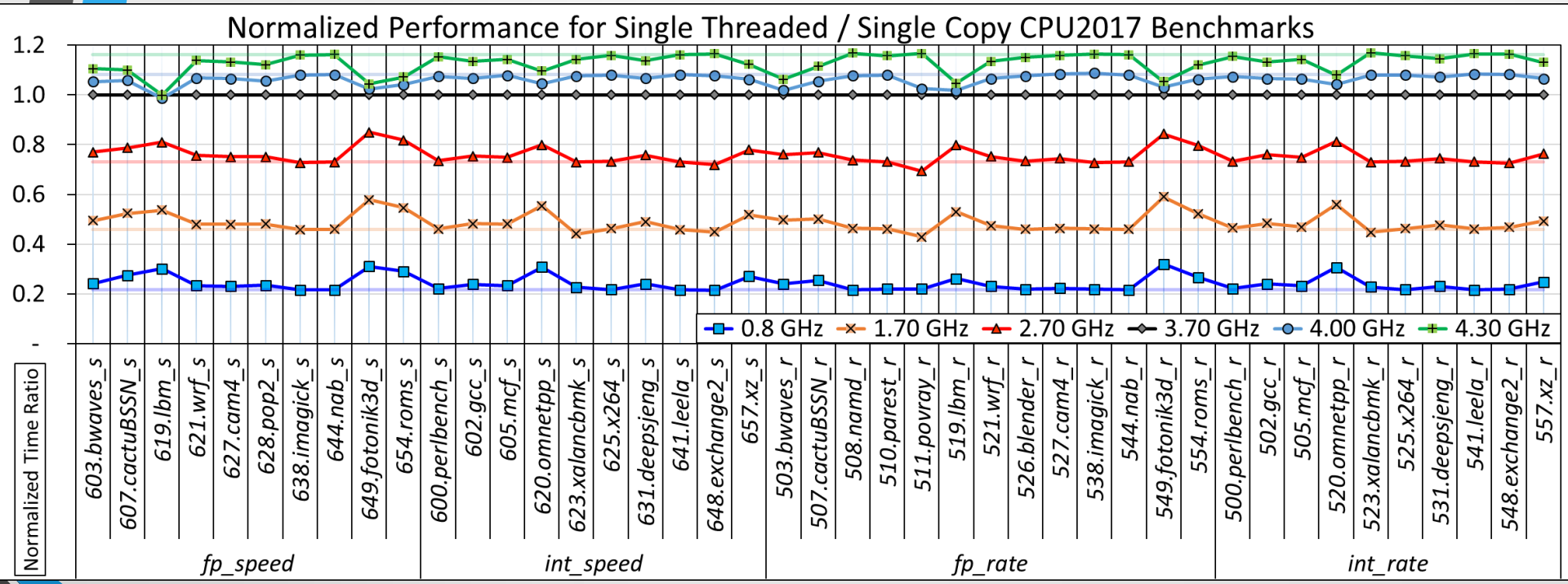
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

Normalized Performance as a Function of Clock Frequency when $\{N_T \mid N_C\}=1$

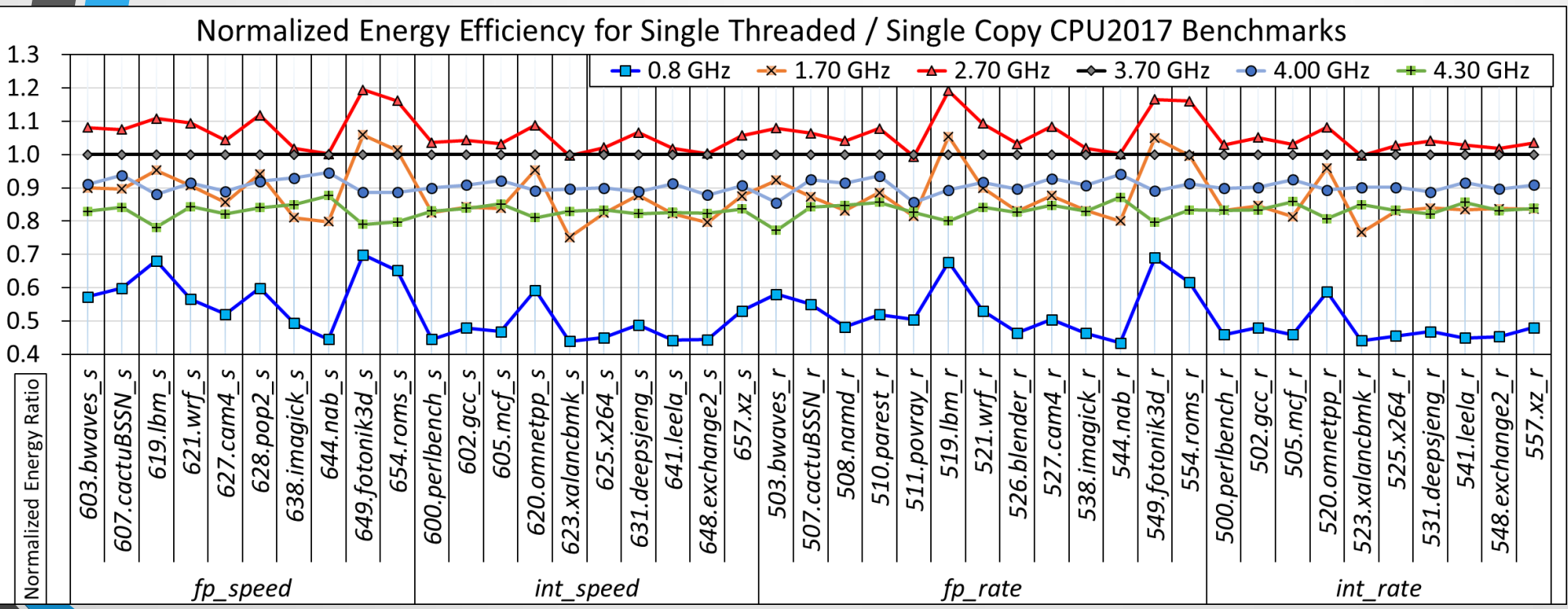
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

Normalized Energy Efficiency as a Function of Clock Frequency when $\{N_T \mid N_C\}=1$

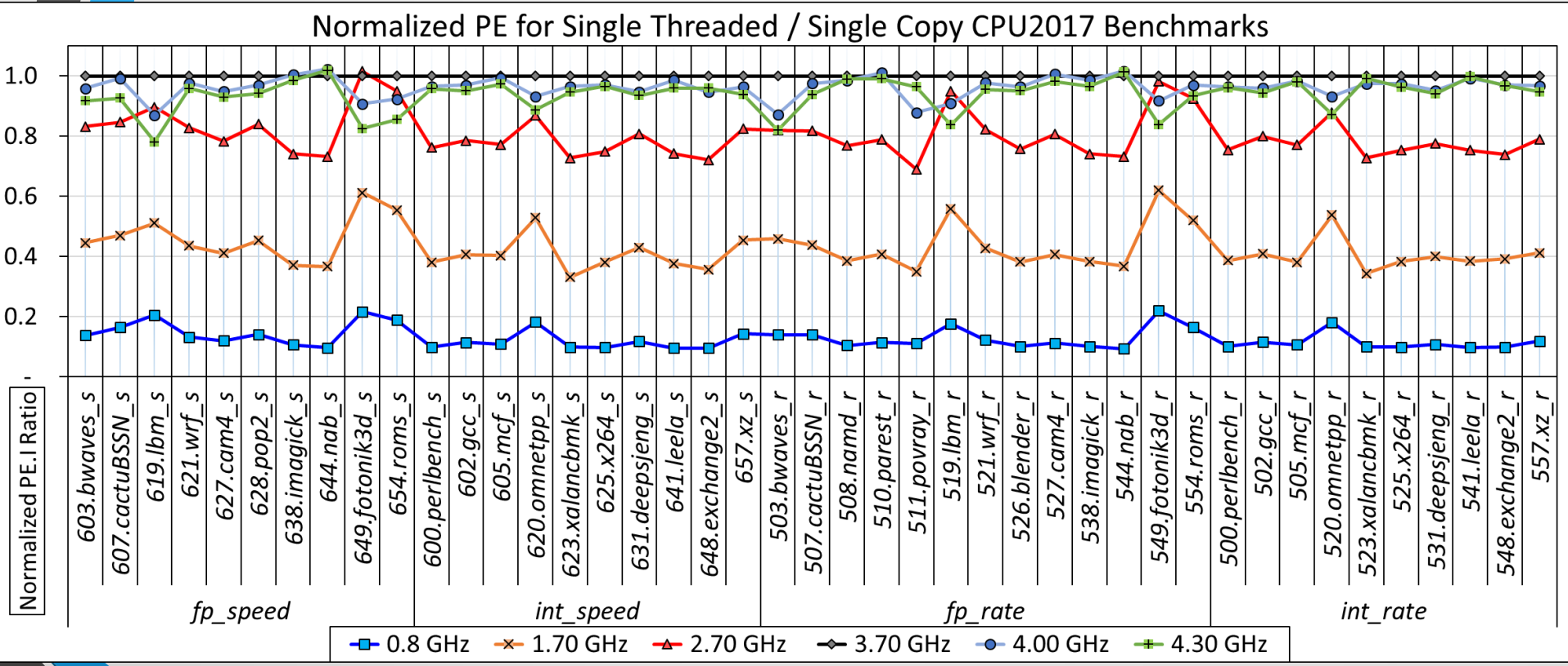
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

Normalized PE as a Function of Clock Frequency when $\{N_T \mid N_C\}=1$

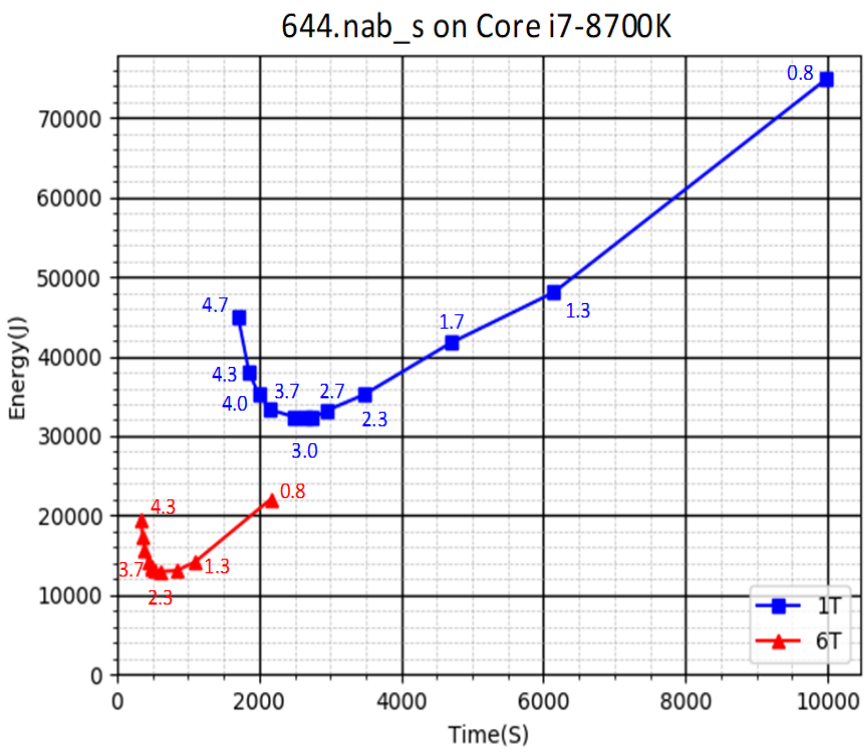
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

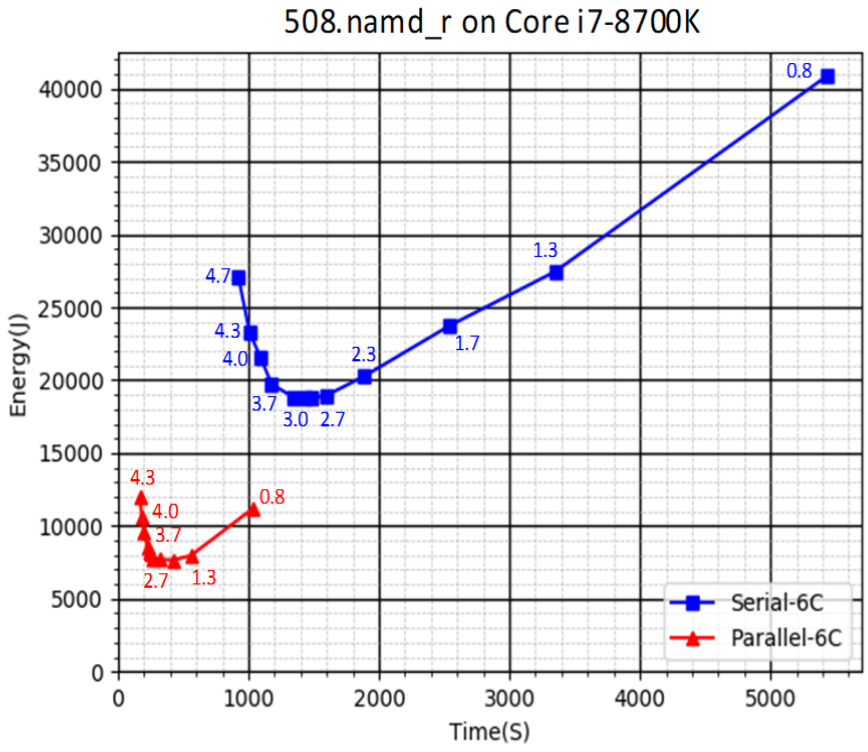


LaCASA

Optimal Configuration Selection

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

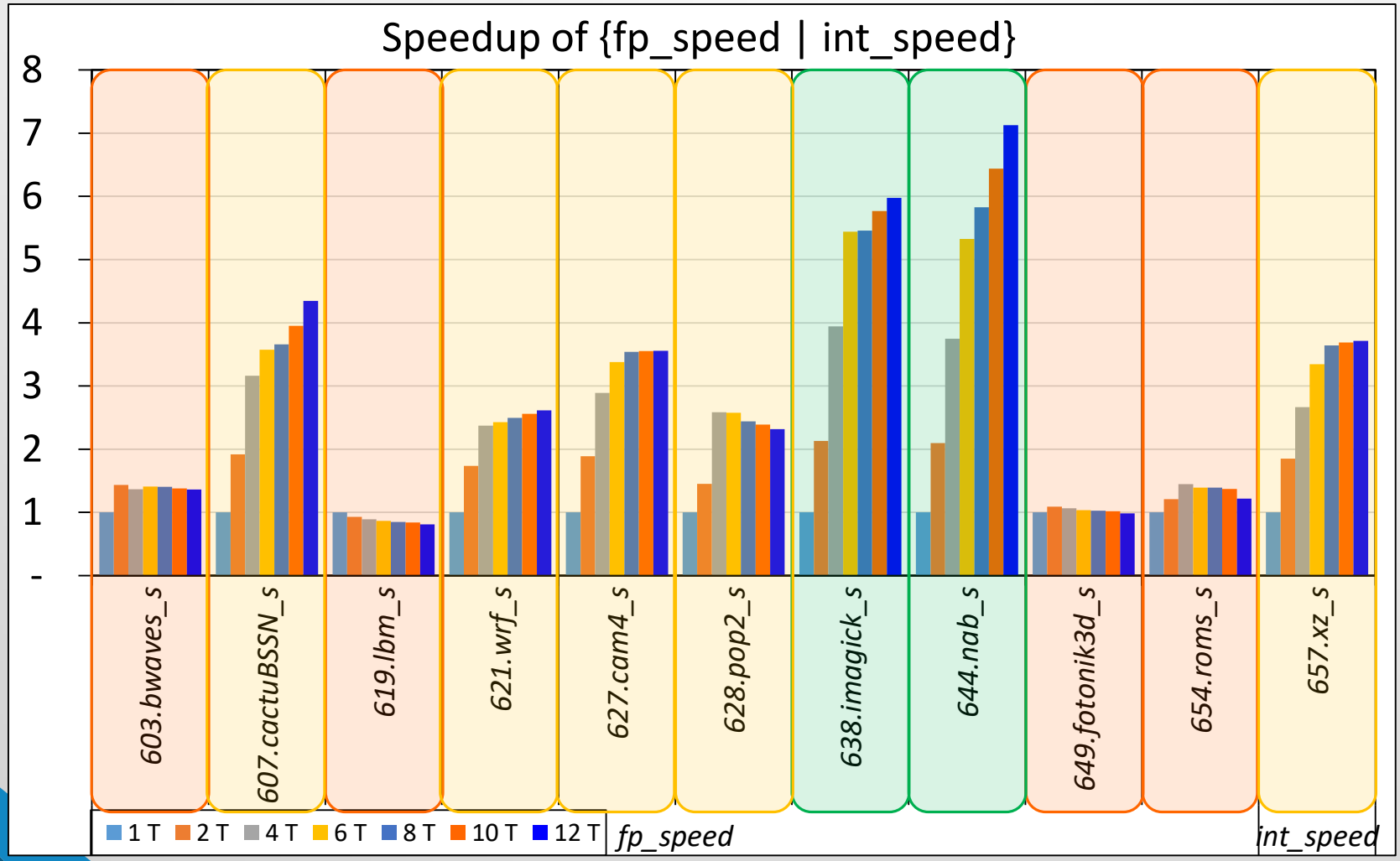




LaCASA

Scalability of Speed Benchmarks

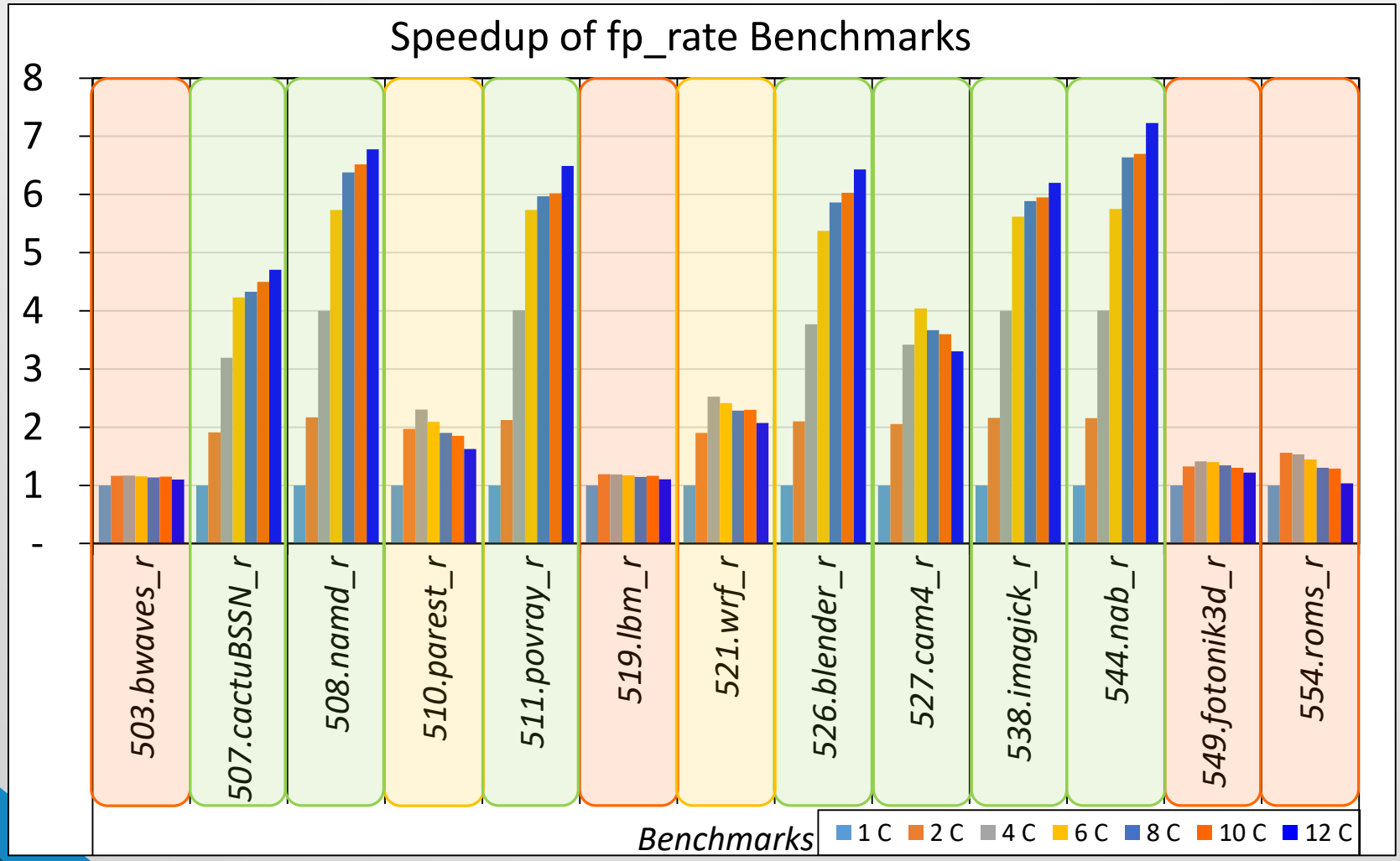
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

Scalability of Floating-Point Rate Benchmarks

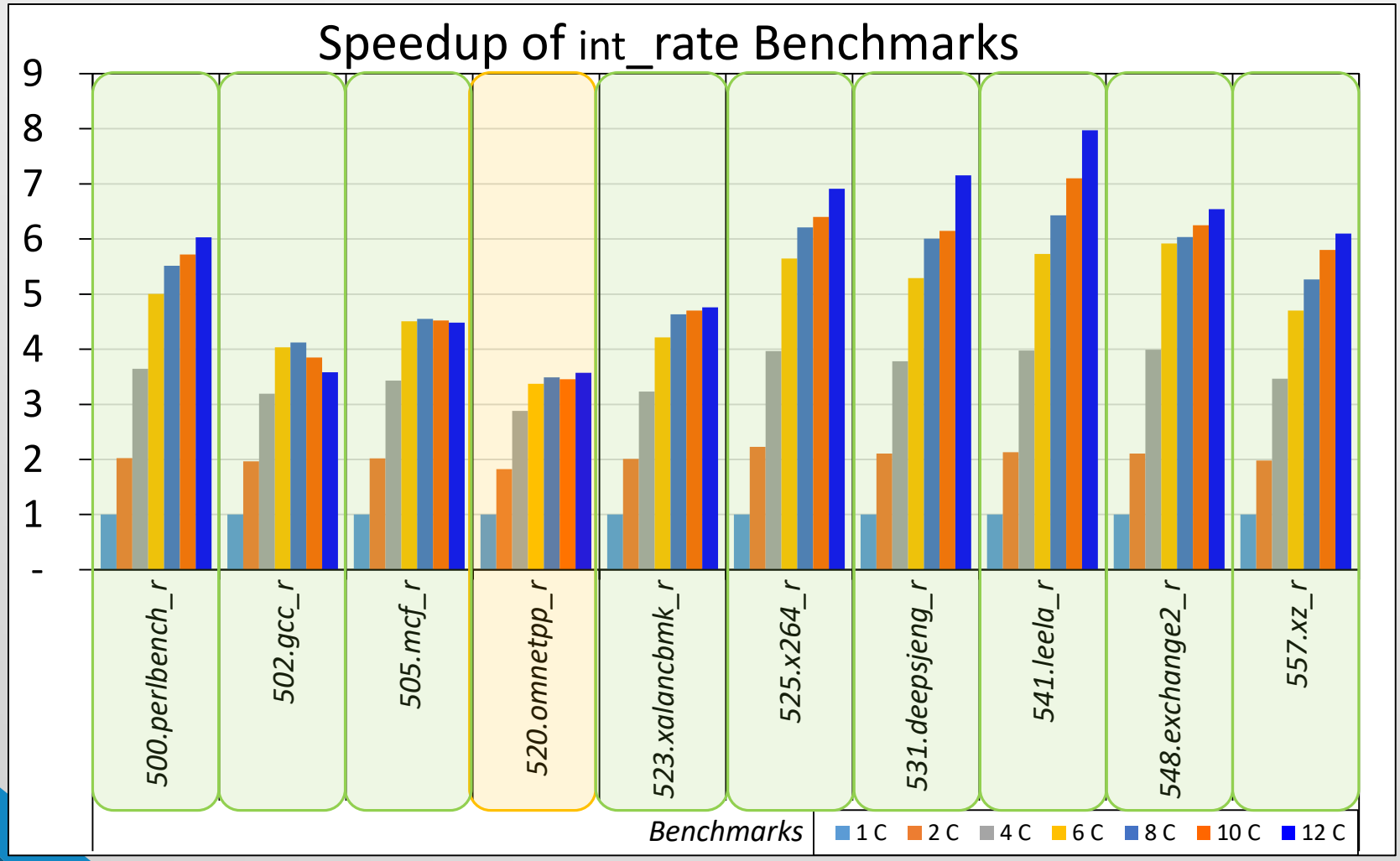
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

Scalability of Integer Rate Benchmarks

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

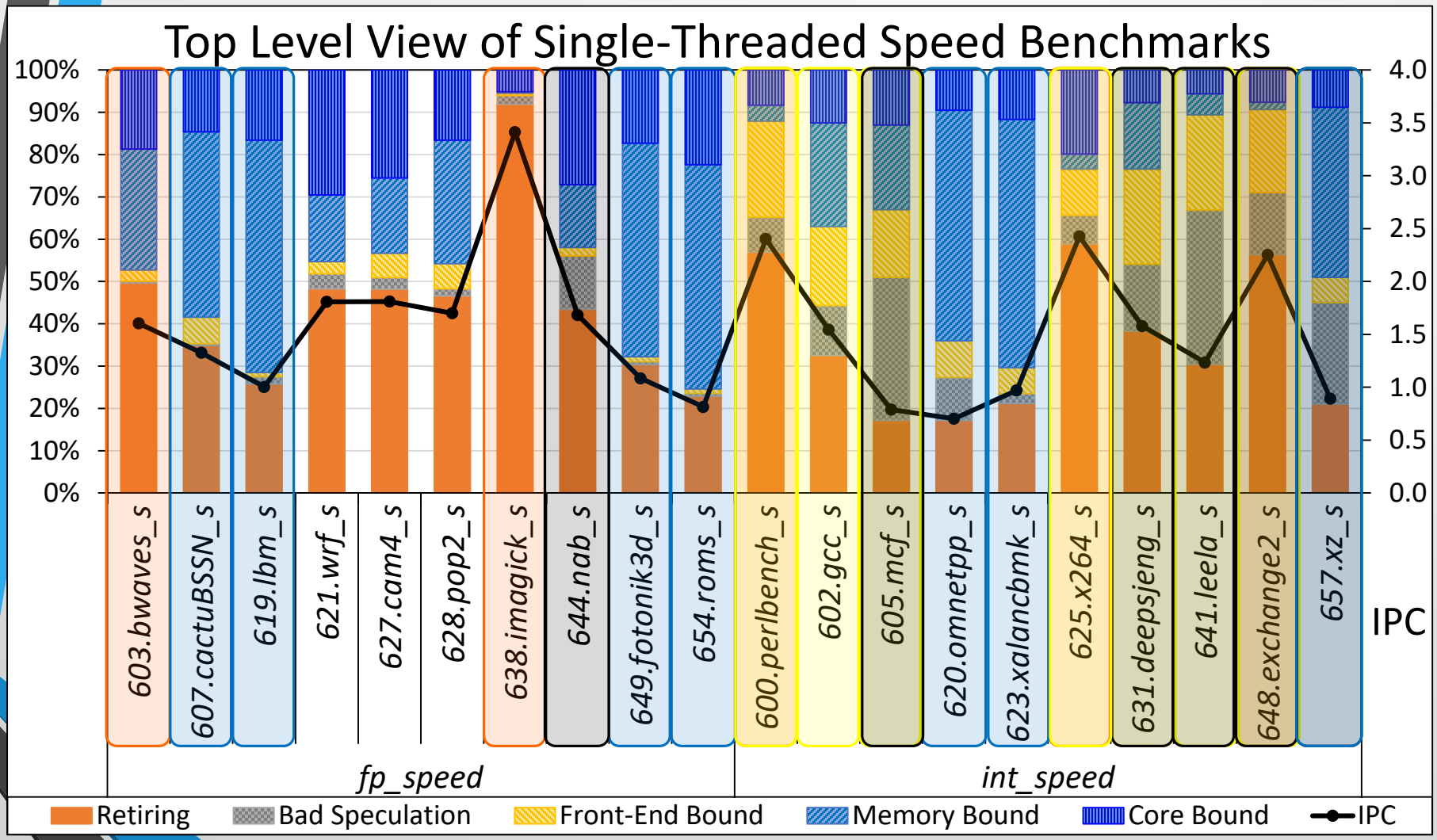
General Parameters for int_speed Benchmarks

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

int_speed	Time [s]	Cycles [Billion]	Instructions [Billion]	Branches [Billion]	Loads [Billion]	Stores [Billion]	IPC
600.perlbench_s_a	108.1	505.88	1,311.37	258.98	359.00	247.62	2.59
600.perlbench_s_b	66.0	308.21	716.17	153.58	232.07	123.60	2.32
600.perlbench_s_c	68.1	318.67	713.62	142.61	221.23	131.98	2.24
602.gcc_s_a	194.2	907.80	1,468.41	409.75	483.08	71.77	1.62
602.gcc_s_b	80.5	375.51	548.41	127.55	153.01	73.49	1.46
602.gcc_s_c	76.5	357.57	532.24	125.25	149.49	69.65	1.49
605.mcf_s	320.5	1,499.32	1,193.65	288.87	422.47	91.84	0.80
620.omnetpp_s	299.0	1,399.21	1,101.10	242.71	372.28	190.77	0.79
623.xalancbmk_s	211.6	989.48	964.65	230.12	273.35	57.18	0.97
625.x264_s_a	16.7	77.86	181.92	13.31	38.06	16.81	2.34
625.x264_s_b	49.6	231.76	570.47	45.59	113.93	44.20	2.46
625.x264_s_c	53.2	247.95	604.38	48.19	124.94	48.73	2.44
631.deepsjeng_s	218.5	1,022.12	1,777.00	230.38	428.05	186.21	1.74
641.leela_s	332.8	1,557.87	1,927.53	298.07	510.41	181.13	1.24
648.exchange2_s	195.3	913.54	2,062.57	233.34	750.13	462.04	2.26
657.xz_s_a	721.4	3,377.22	4,720.70	732.82	1,147.87	351.72	1.40
657.xz_s_b	953.8	4,464.23	3,002.84	445.58	723.61	260.95	0.67

Intel Top-down Characterization for Single-Threaded Speed Benchmarks

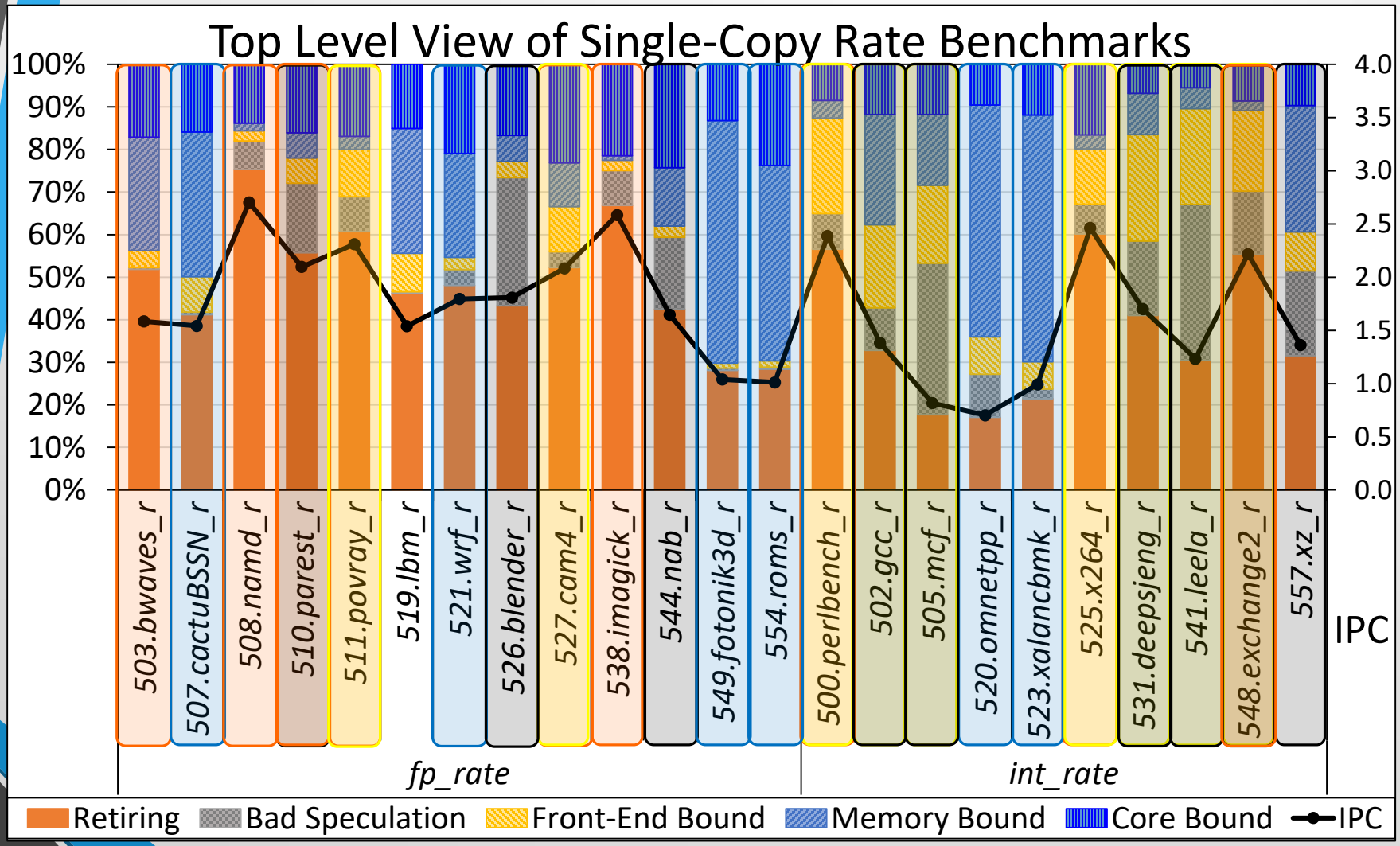
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE



LaCASA

Intel Top-down Characterization for Six-Threaded Speed Benchmarks

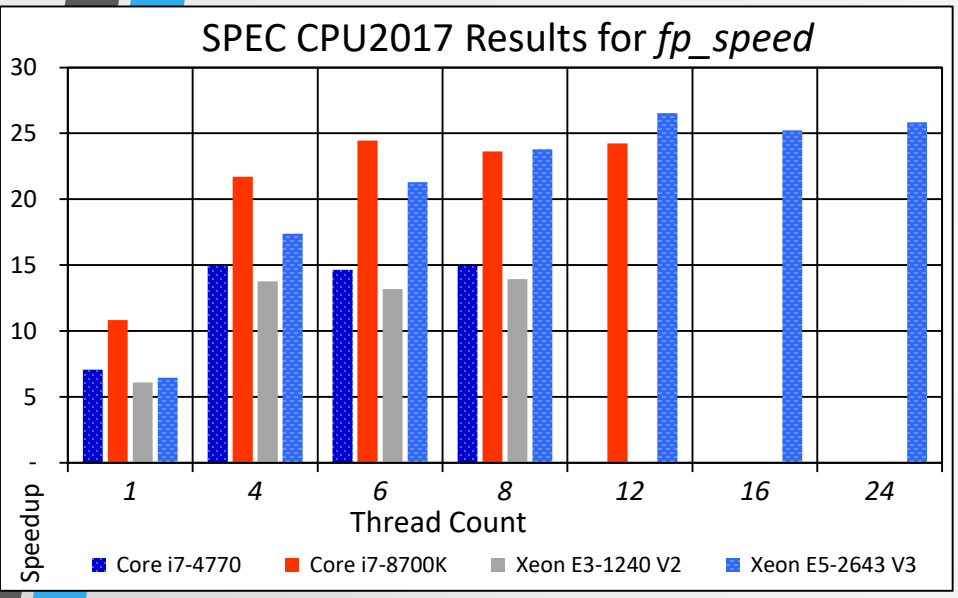
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

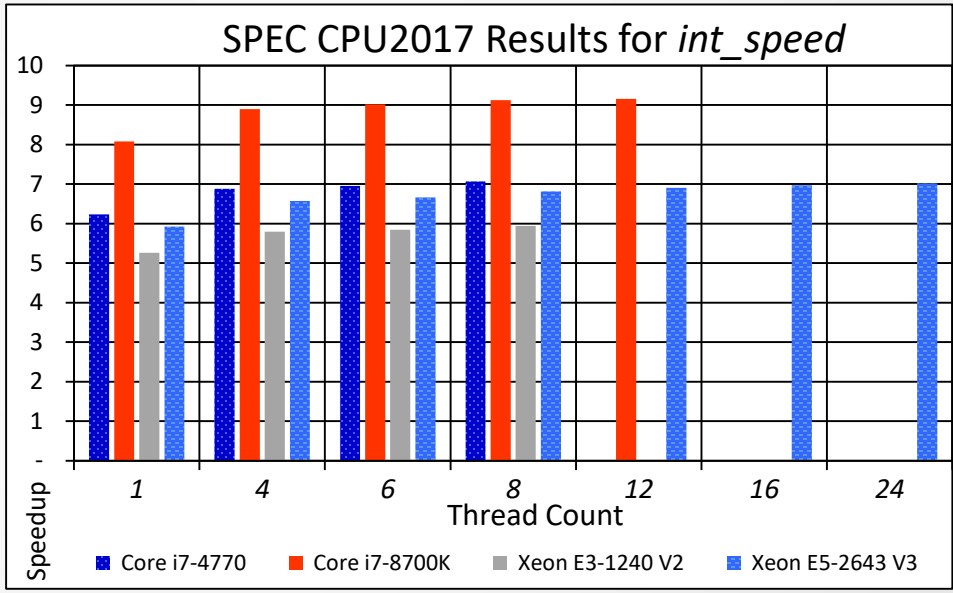


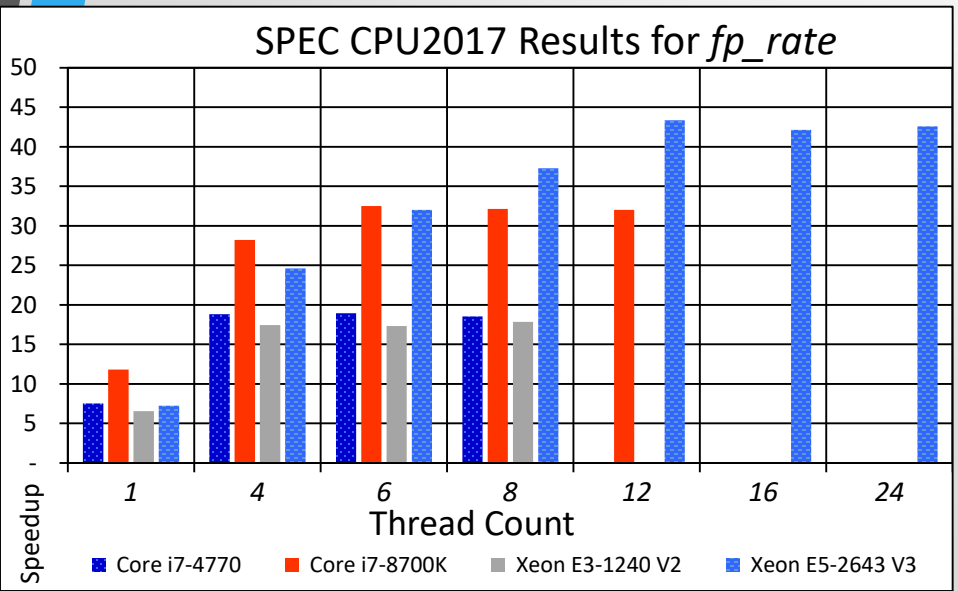
LaCASA

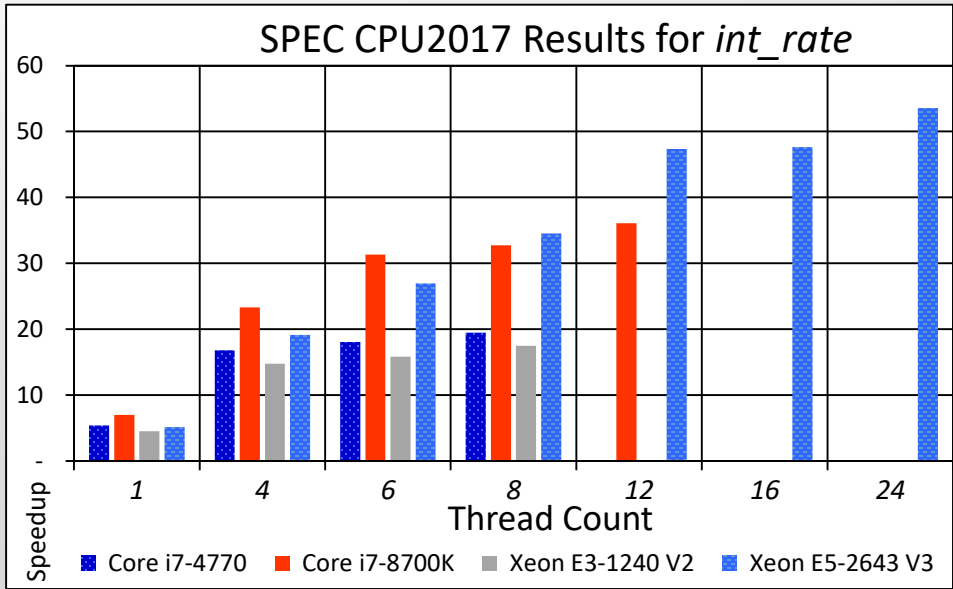
SPEC Numbers Across all Test Systems

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE









Conclusions

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- The latest incarnation of SPEC benchmarks, SPEC CPU2017 is a diverse, real-life applications based benchmark suite that can qualitatively and quantitively evaluate a systems speed and throughput
- IPS executables run faster than GNU executables
- Hardware-supported prefetching is very effective in the single-threaded benchmarks. Disabling hardware prefetching significantly reduces the overall performance. Its impact on performance decreases as the number of threads/copies increase.
- For a single-threaded/copy runs, the best performance is achieved when the clock frequency is at its maximum with turbo mode enabled. The best energy efficiency is achieved for an intermediate processor clock. The best PE is achieved for the nominal frequency.
- For multithreaded/multicopy runs, the best performance is achieved when the clock frequency is at its maximum with turbo mode enabled. However, the best energy efficiency is achieved when the clock frequency is at an intermediate. Finally, the best PE is also obtained when the clock frequency is set to intermediate values.

Conclusion

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- Benchmark Characterization Findings
  - The findings from the Top-down Microarchitectural Analysis Method are as follows. In general, fp_speed benchmarks are mainly bound by stalls in the back-end. int_speed benchmarks are mainly bound by stalls in the front-end and bad speculation.
  - A combined metric called PE.I is introduced to evaluate the scalability of benchmarks when both performance and energy efficiency are considered together
- Scalability Findings
  - Increasing the number of threads/copies to match the number of cores generally improves performance of SPEC CPU2017 benchmarks
  - The fp_speed benchmarks scale well, as long as the number of threads does not exceed the number of physical cores.
  - The rate benchmarks scale well as the number of copies increase up to the number of physical cores. The results indicate that the fp_rate benchmarks do not benefit from hyper-threading, whereas int_rate benchmarks see steady performance improvements.
  - Analyzing performance across different machines, the following observations can be made. (a) Clock frequency is the biggest driving force in performance improvements. (b) The performance of the rate benchmarks is heavily affected by the size of last level cache.

LaCASA

To Learn More

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

- Ranjan's thesis: http://www.ece.uah.edu/~milenka/docs/ranjan.hebbar.thesis.pdf
- Ranjan's ICPE paper: Ranjan Hebbar Seethur Raviraj and Aleksandar Milenkovic, "**SPEC CPU2017: Performance, Event, and Energy Characterization on the Core i7-8700K**," in the *Proceedings of the 10-the ACM/SPEC International Conference on Performance Engineering (ICPE 2019)*, Mumbai, India, April 7-11, 2019, 8 pages. doi: 10.1145/3297663.3310314.
- Ranjan Hebbar Seethur Raviraj and Aleksandar Milenkovic, "**Impact of Thread and Frequency Scaling on Performance and Energy Efficiency: An Evaluation of Core i7-8700K Using SPEC CPU2017**," in the *Proceedings of the IEEE Southeast Con*, Huntsville, AL, April 11-14, 2019, 7 pages.
- Ranjan Hebbar Seethur Raviraj, Mounika Ponugoti, and Aleksandar Milenkovic, "**Battle of Compilers: An Experimental Evaluation Using SPEC CPU2017**," in the *Proceedings of the IEEE Southeast Con*, Huntsville, AL, April 11-14, 2019, 8 pages.

LaCASA

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

Thank You!

Questions?

LaCASA