ARMv8 Micro-architectural Design Space Exploration for High Performance Computing using Fractional Factorial

Roxana Rusitoru
Systems Research Engineer, ARM

Motivation & background

- Goal:
  - HPC-oriented core (characteristics suitable for HPC)
- Why:
  - ARM's main focus has been mobile – we have little knowledge of what an ARM HPC core should look like
- Who:
  - ARM and partners can make more informed decisions if we/they are to create an HPC-oriented core
- How (first step):
  - Use fractional-factorial experimental design to explore micro-architectural features*
  - HPC mini-applications & benchmarks
  - Single core, single thread experiments

* Previously used by Dam Sunwoo et al in "A Structured Approach to the Simulation, Analysis and Characterization of Smartphone Applications"

This study


ARMv8 Micro-architectural Design Space Exploration for High Performance Computing using Fractional Factorial

Roxana Rusitoru
ARM Ltd.
110 Fulbourn Road
Cambridge, CB1 9NJ, UK
roxana.rusitoru@arm.com

ABSTRACT

Categories and Subject Descriptors

General Terms

Keywords

1. INTRODUCTION

- This study is…
  - A design space exploration on ARMv8 in-order and out-of-order core configurations to determine the sensitivities of HPC applications with respect to micro-architectural changes.
  - A way to guide detailed micro-architectural investigations (it can point us in the right direction)
- This study is not…
  - A way to produce an "ideal" core configuration that we can just use to create next-gen HPC cores

ARM

Infrastructure background

- gem5
  - Event-based simulator used for computer systems architecture research.
  - Can run full-system simulations, with variable levels of detail.
  - Enables the exploration of various new and existing micro-architectural features, whilst running the same software stack as real hardware.
- SimPoint
  - Provides a mechanism and methodology for extracting the most representative phases from a given workload.
  - Each SimPoint consists of a warm-up period and a region of interest. Their size is given in number of instructions.
- Fractional Factorial
  - Relies on sparsity-of-effects principle (only the main and low-order interactions are investigated).
  - This allows for a significant reduction in the number of experiments (fraction of a full factorial).

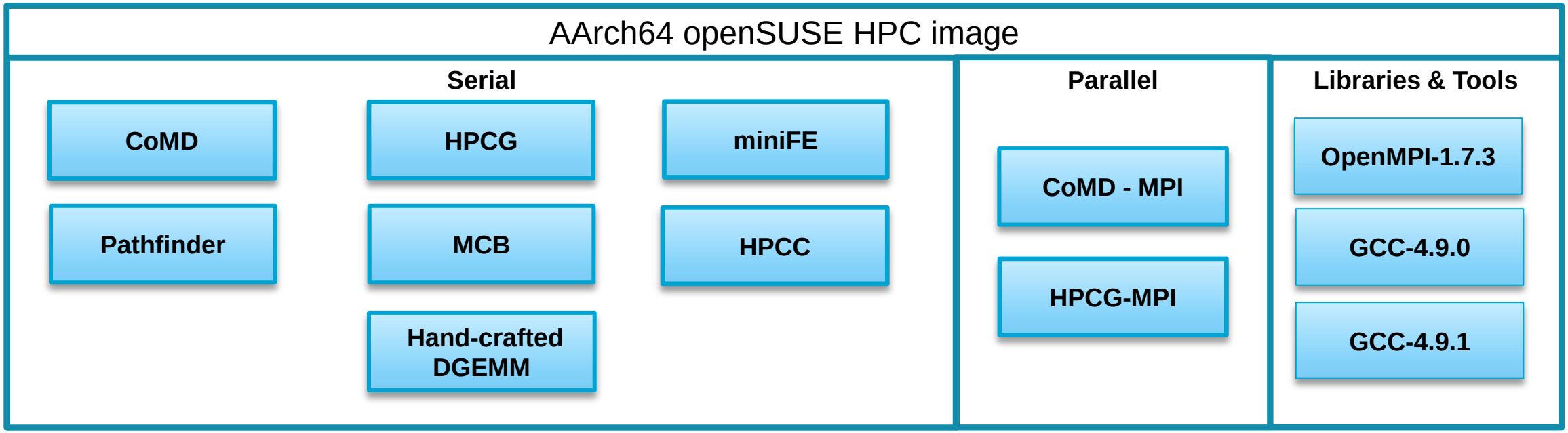
Methodology

- Select a representative collection of HPC proxy applications and benchmarks
- Determine gem5-appropriate runtime parameters for those applications
- Gather and validate SimPoints
- Determine appropriate micro-architectural parameters and values
- Run fractional factorial experiments

- All our experiments are single core, single thread.
- Figure-of-Merit: IPC

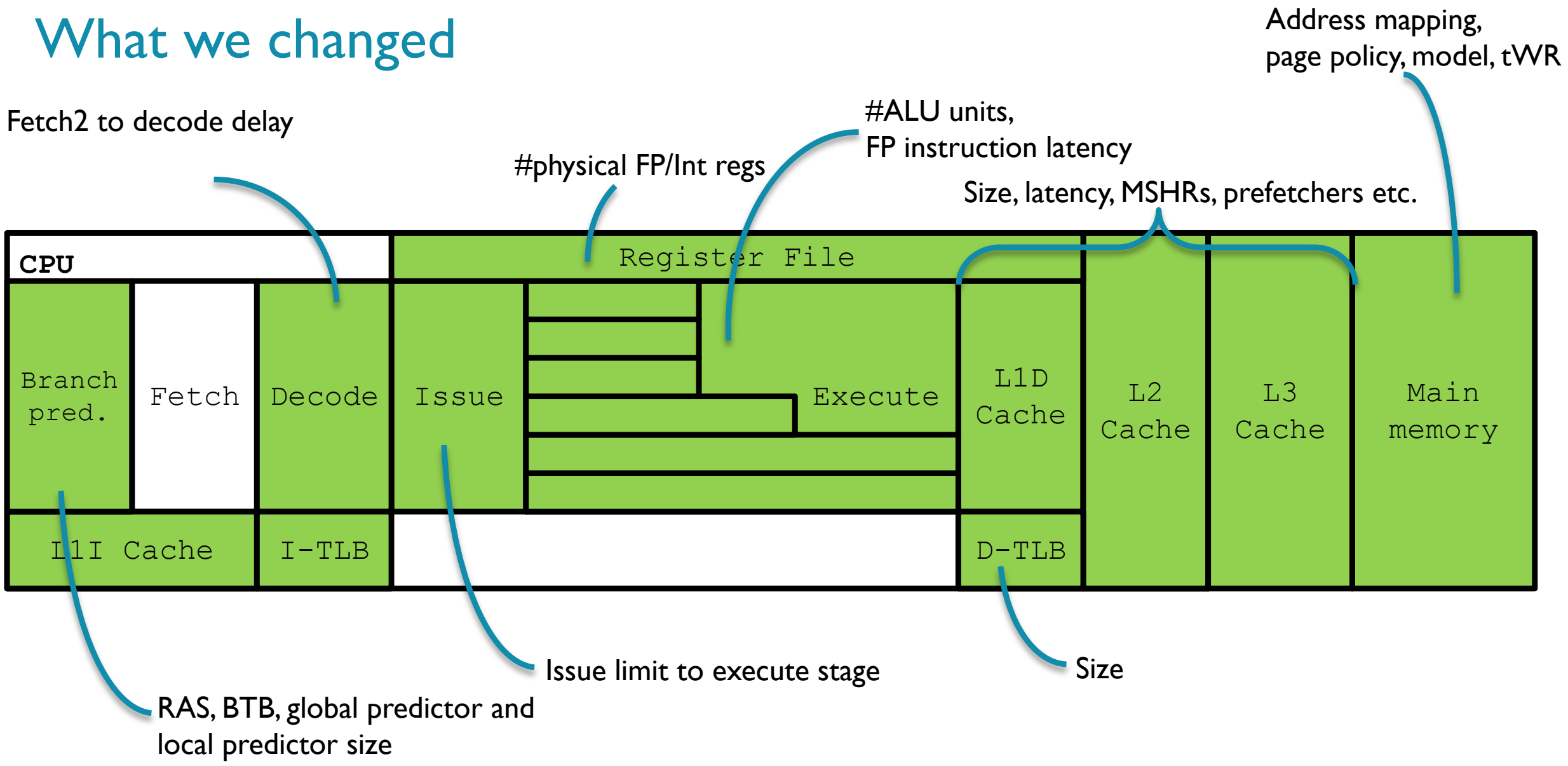
Applications

- We chose problem sizes such that the total memory footprint is larger than the total maximum size of caches.
- For all applications we only ran the core loops.
- For most applications, we used 1B instruction SimPoints with 100M instruction warm-up phases.



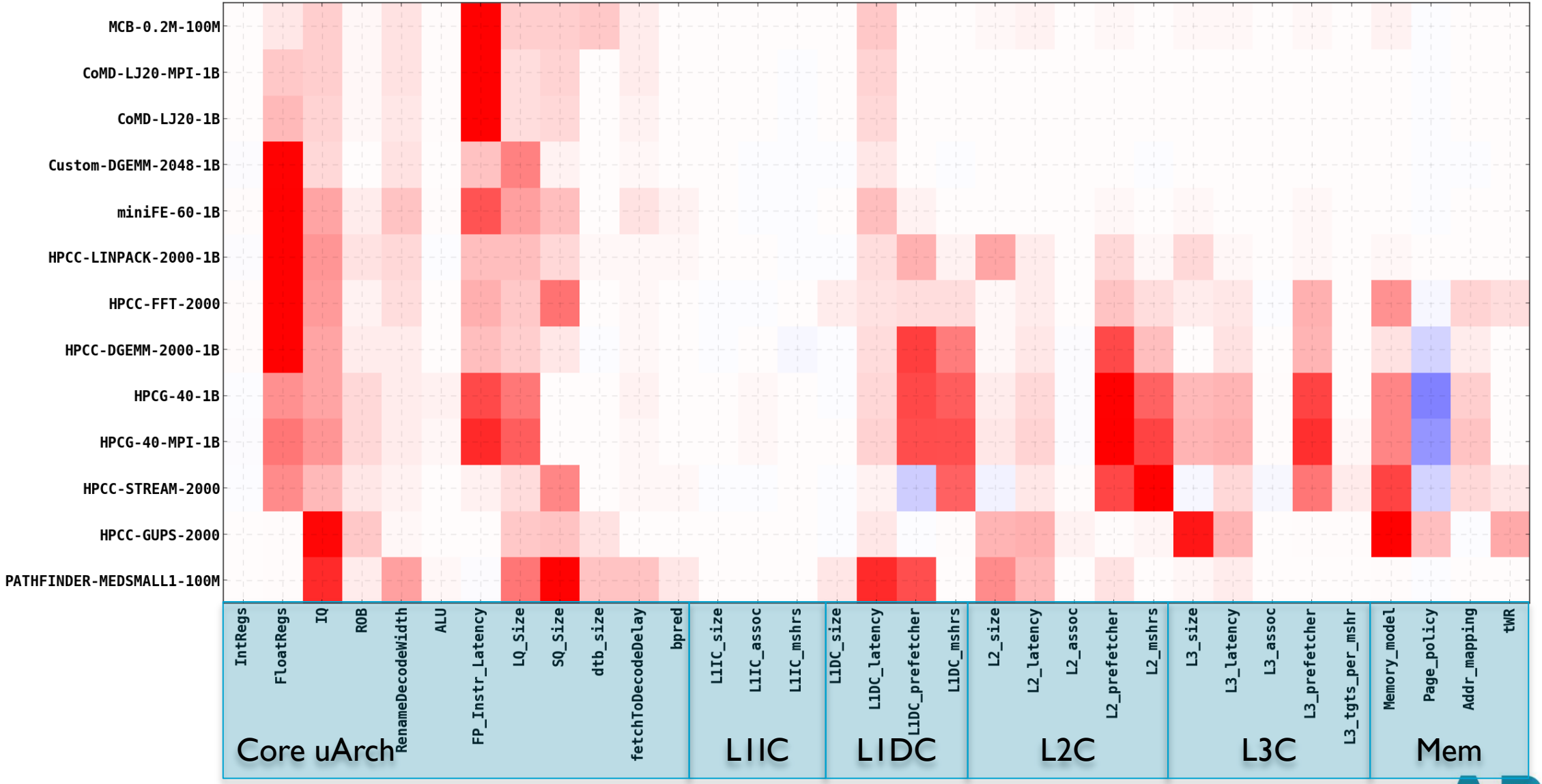
ARM

What we changed

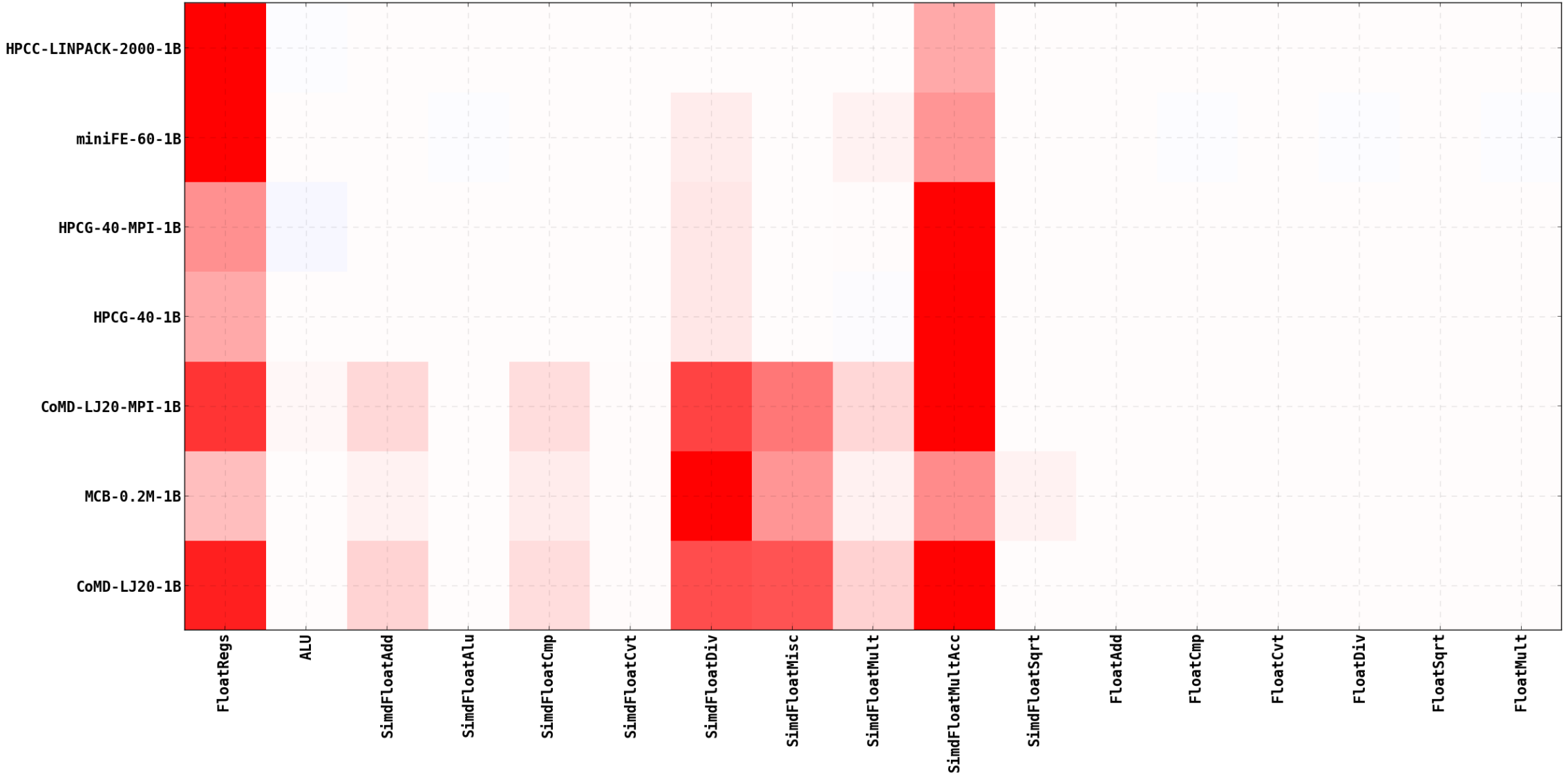


ARM

OoO study – fractional factorial results (ARM Cortex-A57-like model-based)

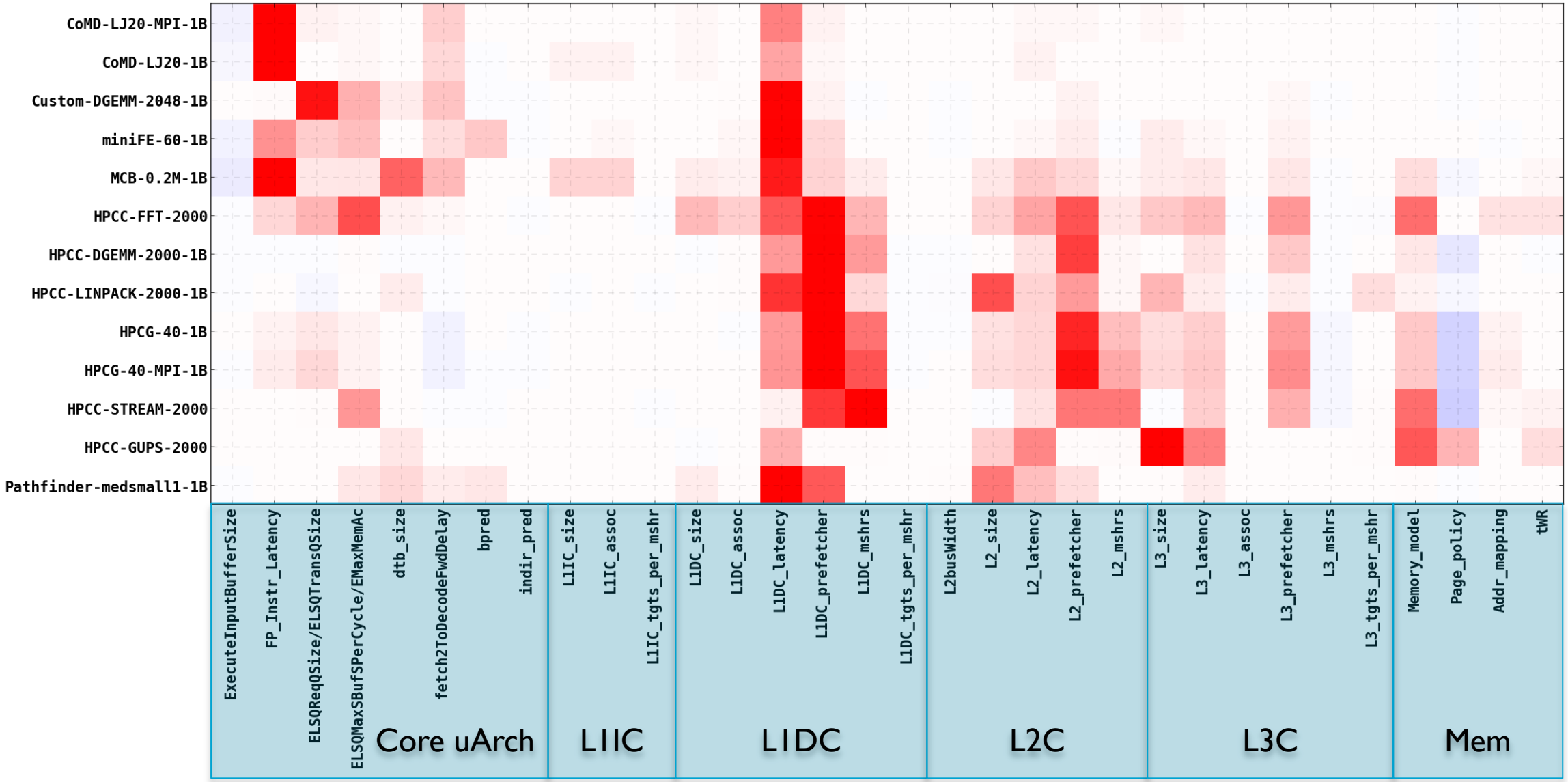


OoO study – floating-point instruction latency



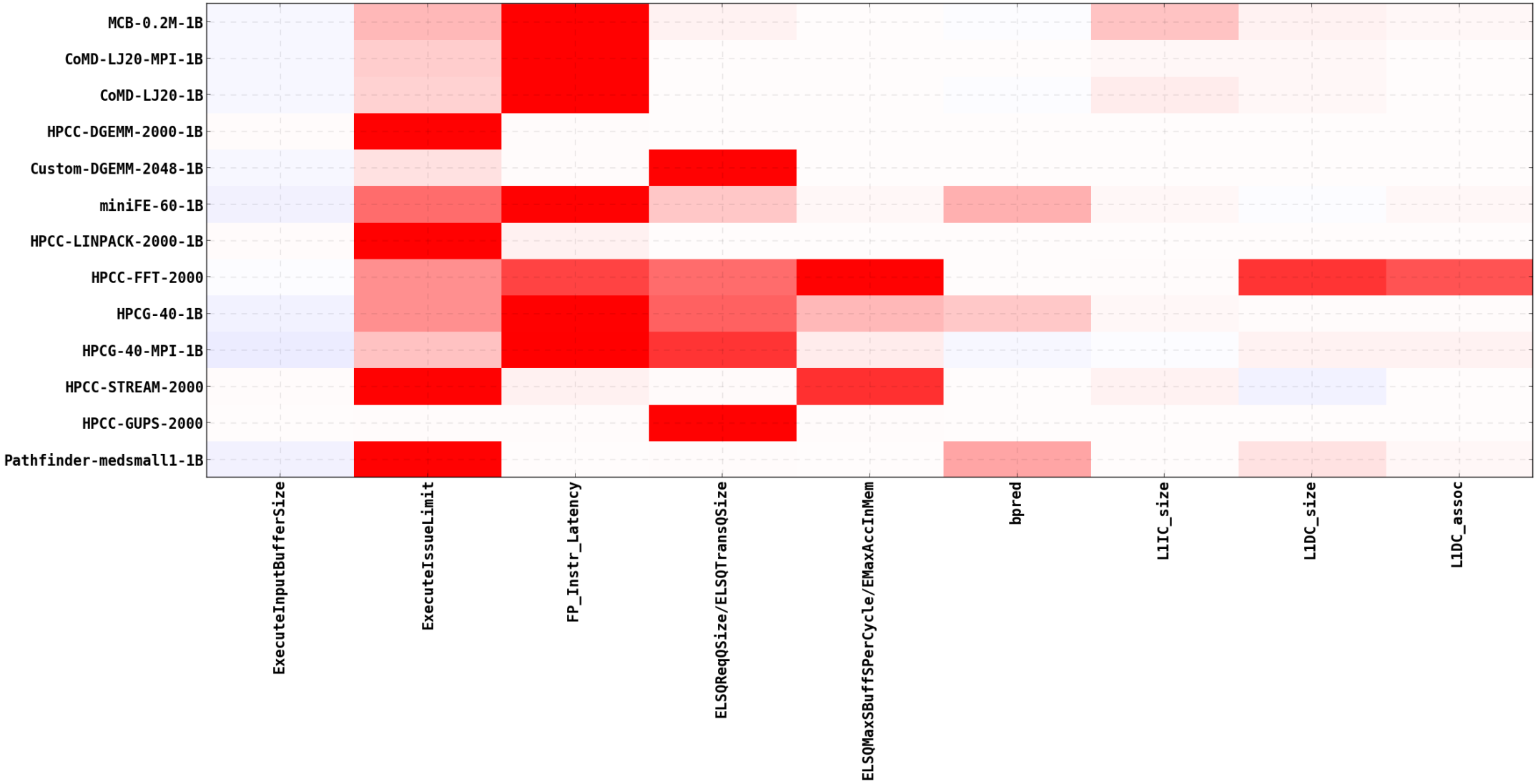
ARM

In-order study – fractional factorial results (ARM Cortex-A53-like model-based)



ARM

In-order study – front end study



ARM

Conclusions

- High sensitivity to latency versus throughput
- For out-of-order cores, there is an increased sensitivity to having more FP physical registers
- For out-of-order cores, there is no sensitivity to an increased number of LD/ST/Int ALUs
- In-order core shows sensitivity towards L1, L2, L3 prefetchers and memory model
- Little or no sensitivity towards L1, L2, L3 data cache size variations
- Negative sensitivity when changing the page policy

Summary

- We investigated single-core configurations of both out-of-order and in-order processors
  - This provided us with a good "within core" perspective
  - Latency, and not throughput, matters most
- Further work:
  - Investigation into data cache size sensitivity
  - In-order core prefetcher investigation (on-going)
- Future studies:
  - Multi-core study using multi-threaded applications (on-going)
  - Deep-dive into the memory system (on-going)
  - SMT study

Future considerations

- We had a methodology in-place for single-core studies, however, is this the best way forward? What about multi-core studies?
  - Methodology (speed/accuracy)
  - Source and magnitude of sensitivities
  - Scalability
  - Figure-of-merit – currently IPC
  - gem5
  - It's easy to go outside of the expected design space. Great for bug hunting, good for pushing the envelope, but is it relevant?

Appendix

ARM

Out-of-order sensitivity study parameters

Parameter	Low	High	Parameter	Low	High
IntRegs	128	160	L1DC_latency	4	2
FloatRegs	192	512	L1DC_prefetcher	NULL	NeighborPrefetcher
IQ	8	64	L1DC_mshrs	4	8
ROB	40	128	L2_size	256KB	1024KB
RenameDecodeWidth	2,2	4,4	L2_latency	10	5
ALU	1,1,1,2	2,2,2,3	L2_assoc	8	16
FP_Instr_Latency	5,4,5,5,64,10,6,10,64,5,6,10,32,32,32,32,6	4,3,3,4,13,3,4,3,12,4,3,4,13,13,12,12,4	L2_prefetcher	NULL	NeighborPrefetcher
			L2_mshrs	4	32
LQ_Size	8	32	L3_size	512KB	4096KB
SQ_Size	8	32	L3_latency	30	20
dtb_size	32	256	L3_assoc	16	32
fetchToDecodeDelay	3	1	L3_prefetcher	NULL	NeighborPrefetcher
bpred	2048,1024,8192,8192,2048,16	8192,4096,32768,32768,8192,64	L3_tgts_per_mshr	8	16
L1IC_size	32KB	64KB	Memory_model	LPDDR2_S4_1066_x32	DDR3_1600_x64
L1IC_assoc	2	8	Page_policy	open_adaptive	closed_adaptive
L1IC_mshrs	2	6	Addr_mapping	RoRaBaChCo	RoCoRaBaCh
L1DC_size	32KB	64KB	tWR	60ns	15ns

ARM

In-order sensitivity study parameters

Parameter	Low	High	Parameter	Low	High
ExecuteInputBufferSize	7	10	L1DC_mshrs	4	8
ExecuteIssueLimit	2	4	L1DC_tgts_per_mshr	8	32
FP_Instr_Latency	2,18,2,2,4,0,2,18,2,2,4,2,2,2,18,18	3,7,4,4,5,2,0,6,4,4,5,4,4,3,7,6	L2busWidth	32	64
ExLSQReqQSize/ExLSQTransQSize	1,2	3,6	L2_size	256KB	1024KB
ExecLSQMaxStoreBuffStorePerCycle/ ExecMaxAccessesInMem	2,2	6,6	L2_latency	10	5
			L2_prefetcher	NULL	NeighborPrefetcher
dtb_size	32	256	L2_mshrs	4	32
fetch2ToDecodeFwdDelay	3	1	L3_size	512KB	4096KB
bpred	2048,1024,8192,8192,2048,16	8192,4096,32768,32768,8192,64	L3_latency	30	20
indir_pred	False	True	L3_assoc	16	32
L1IC_size	32KB	64KB	L3_prefetcher	NULL	NeighborPrefetcher
L1IC_assoc	2	8	L3_mshrs	16	32
L1IC_tgts_per_mshr	8	16	L3_tgts_per_mshr	8	16
L1DC_size	32KB	64KB	Memory_model	LPDDR2_S4_1066_x32	DDR3_1600_x64
L1DC_assoc	2	4	Page_policy	open_adaptive	closed_adaptive
L1DC_latency	4	2	Addr_mapping	RoRaBaChCo	RoCoRaBaCh
L1DC_prefetcher	NULL	NeighborPrefetcher	tWR	60ns	15ns

ARM