Seminar in computer architecture

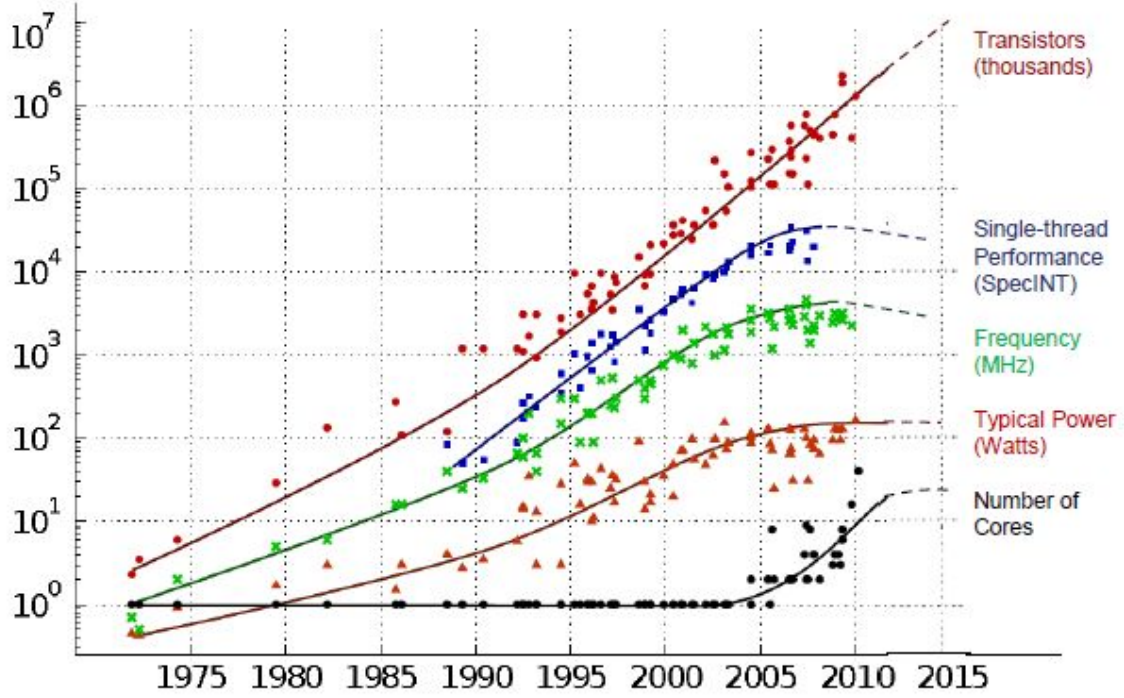
The Case for a Single-Chip Multiprocessor

Kunle Olukotun, Basem A. Nayfeh, Lance Hammond, Ken Wilson, and Kunyung Chang

ASPLOS 1996

Presented by Gauthier de Chezelles

Background: We are in 1996



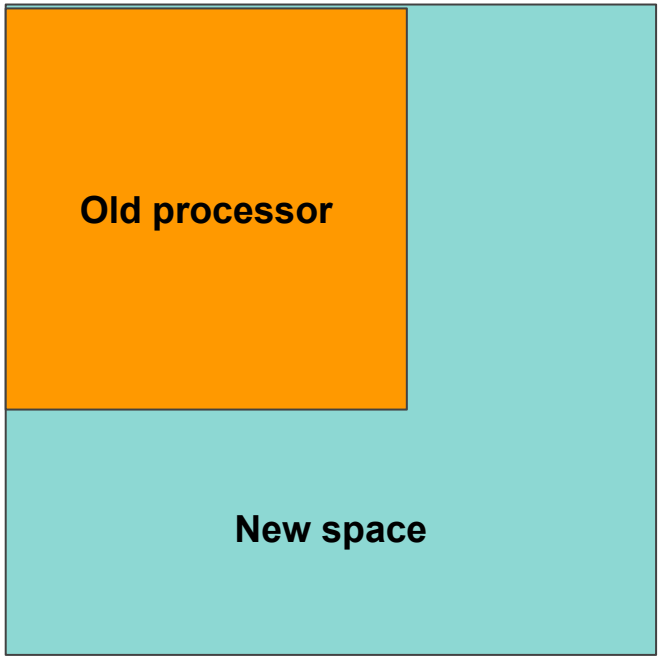
- Number of transistors per chip increases
- Frequency increases
- Number of cores is constant

Executive summary

- Background: The size reduction of transistors gives us opportunities to innovate
- Problem: Increasing the number of issues in a superscalar architecture is not sustainable
- Goal: Design a simpler processor with multiple smaller CPUs
- Key contributions:
  - **Demonstration:** Proves that superscalar architectures are not scalable
  - **Innovation:** Designs and compares a superscalar architecture and a multiprocessor architecture
  - **Interpretation:** Identifies different types of applications and compares their performances on each architecture

Background

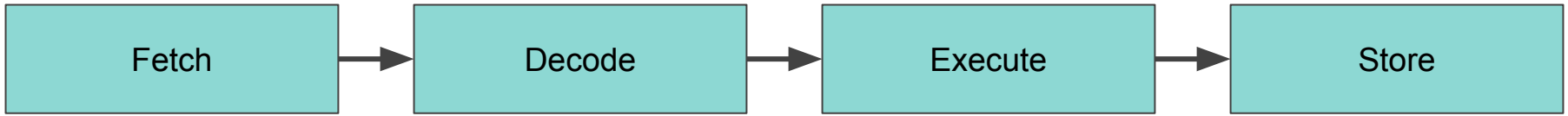
We have more space. What do we do?



- The trend is to do **superscalar architectures**
- This paper proposes it to a **single-chip multiprocessor**

Lifetime of an instruction

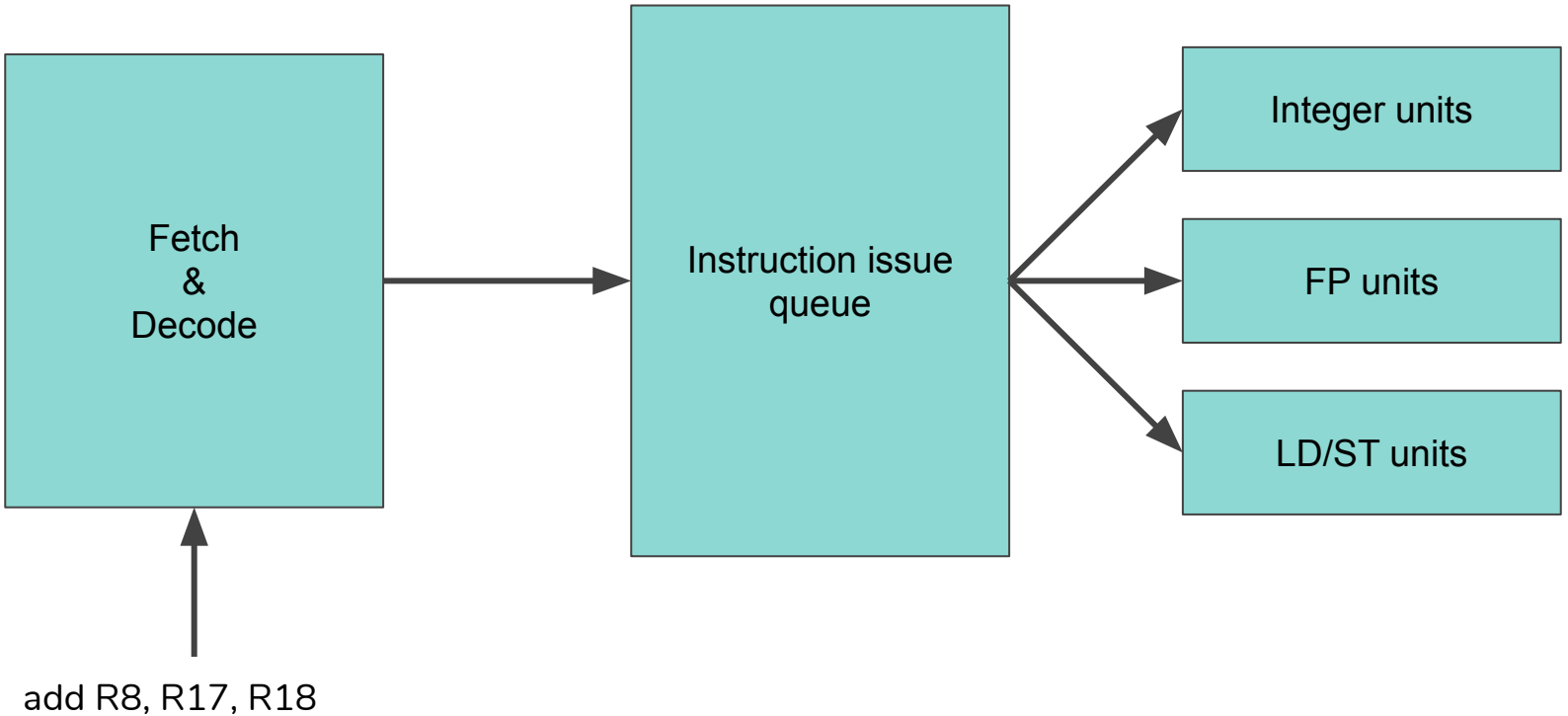
add R8, R17, R18



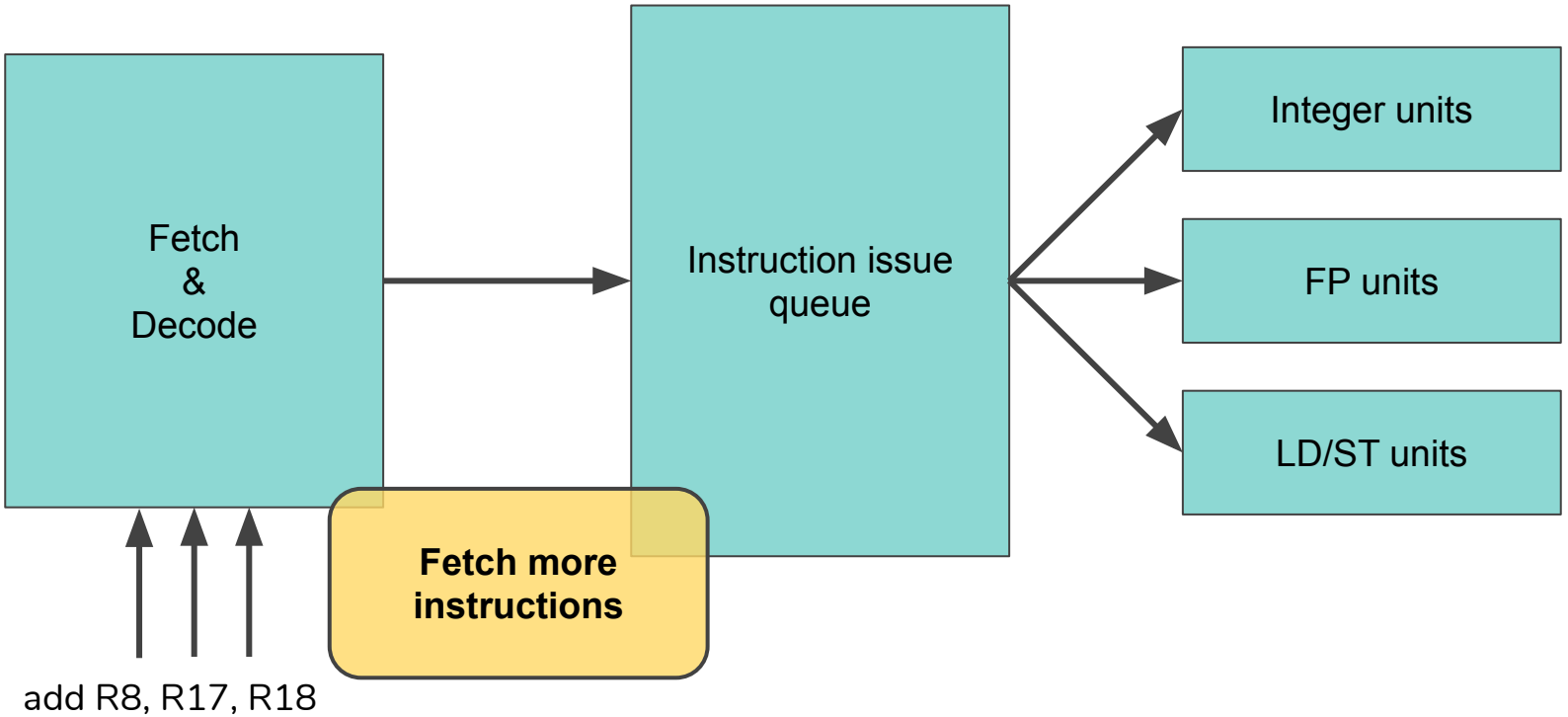
How can we make this faster ?

- Increase frequency
  - Limited
- We can do more pipelining
- Can we do even better ?

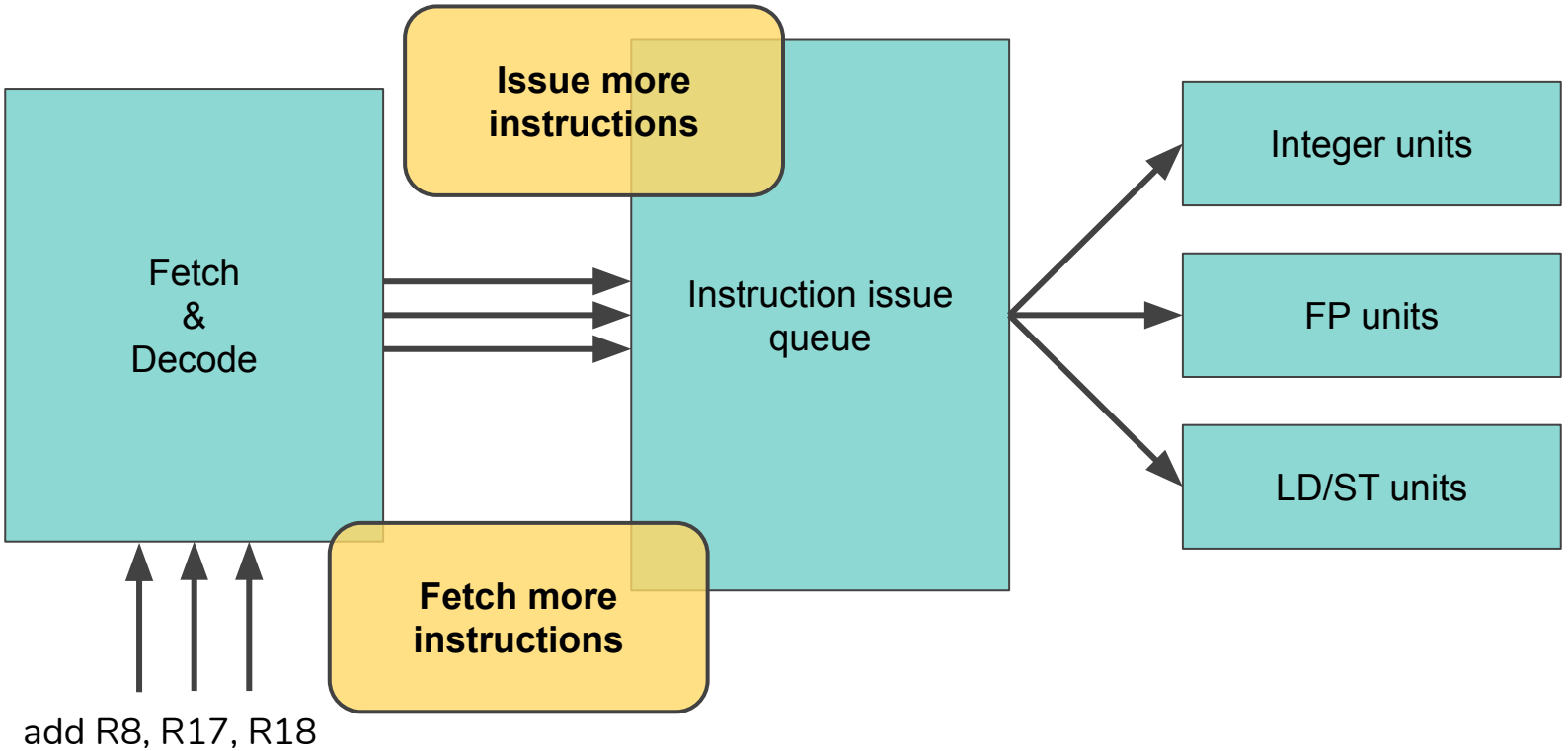
Superscalar architecture



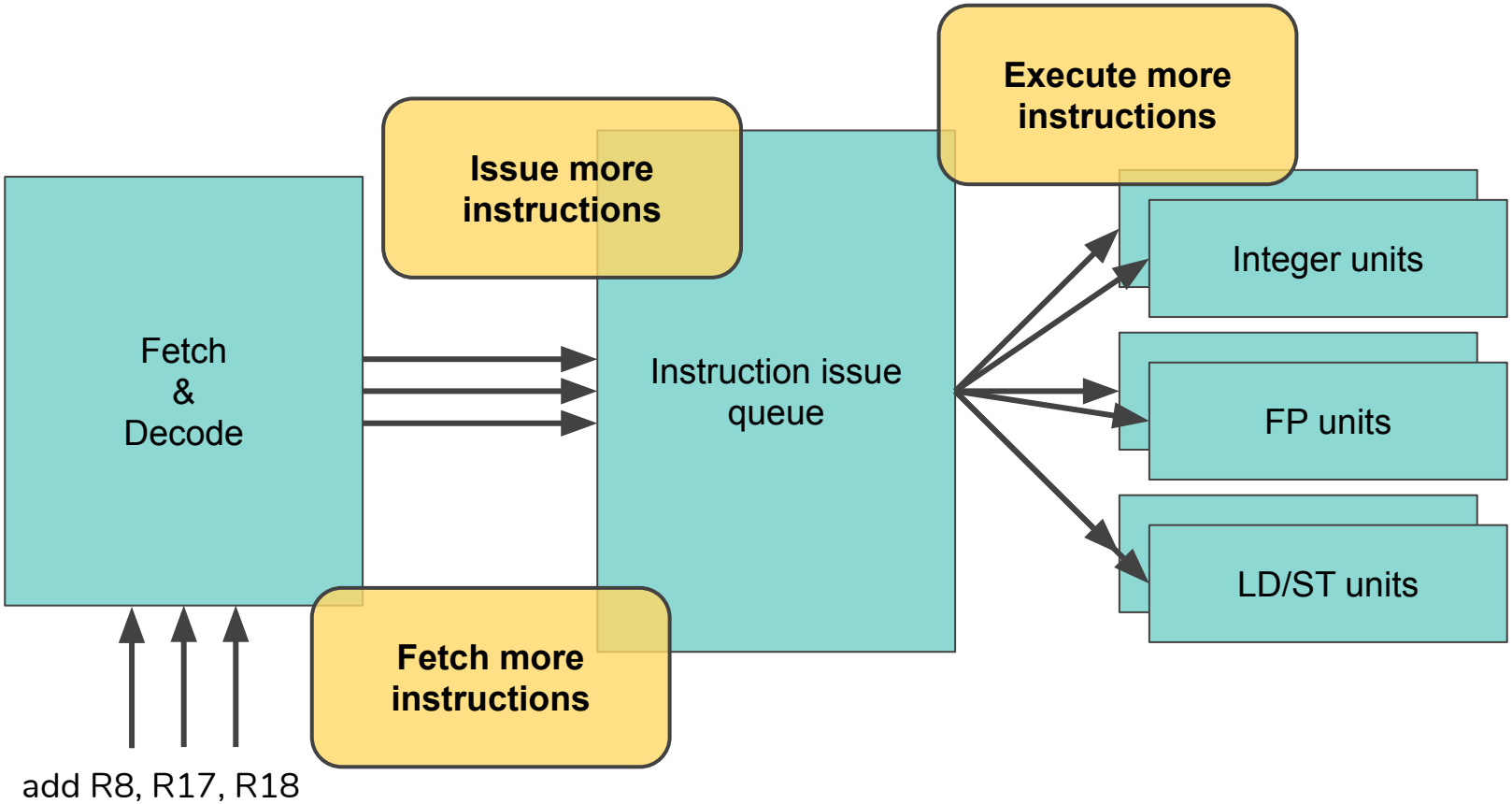
Superscalar architecture



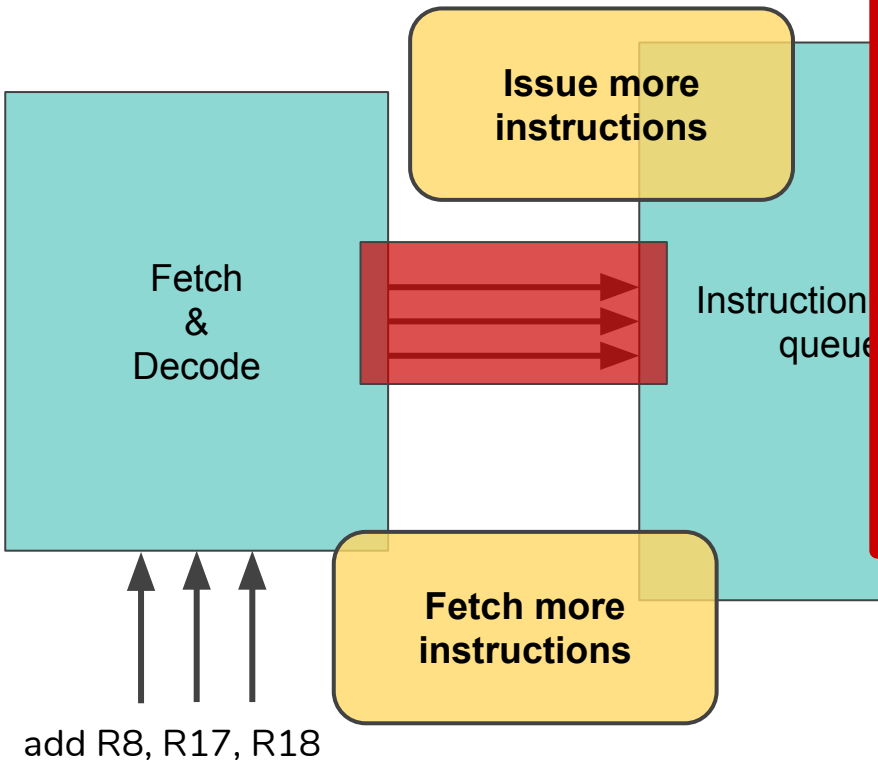
Superscalar architecture



Superscalar architecture



Superscalar architecture



VOCABULARY

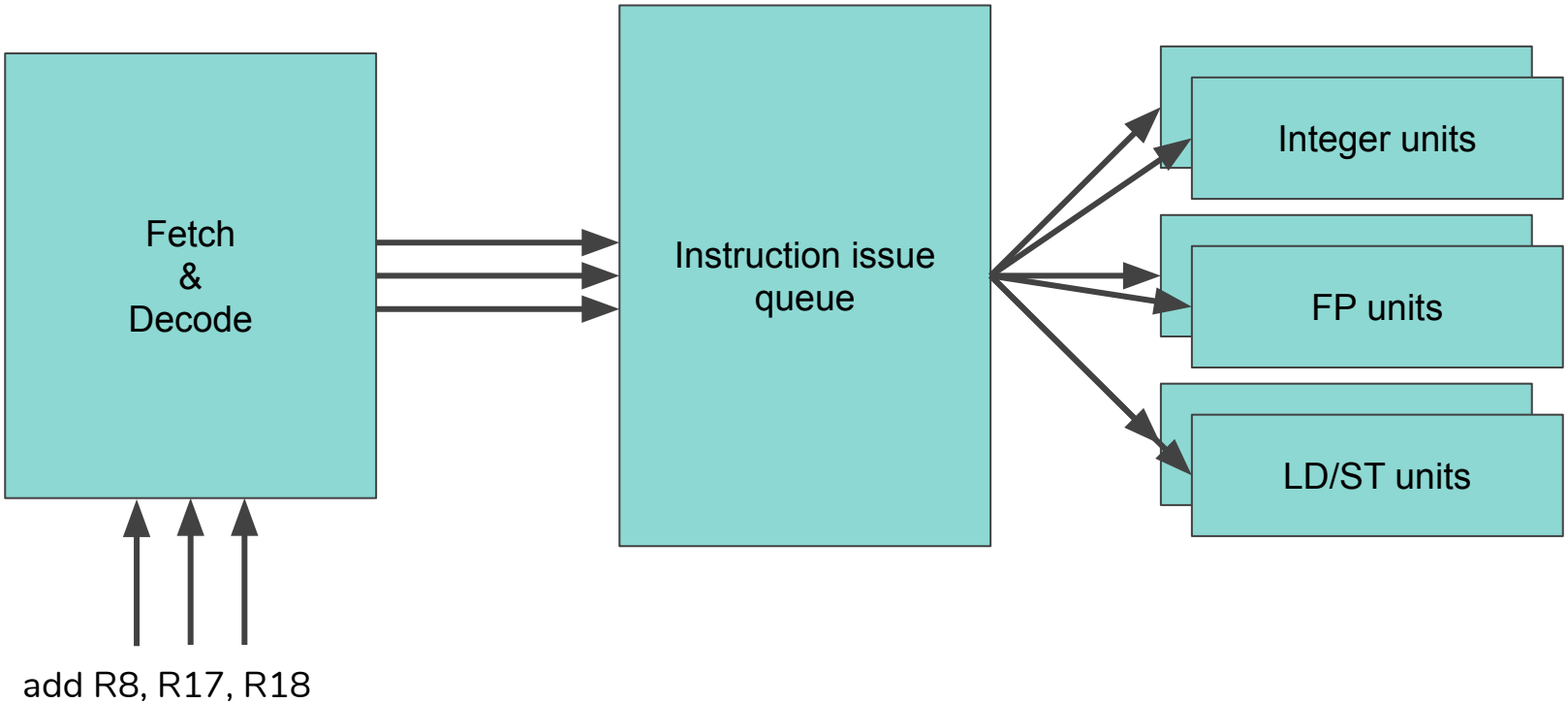
The **issue width** is the number of instructions **issued** per cycle

EXEMPLE

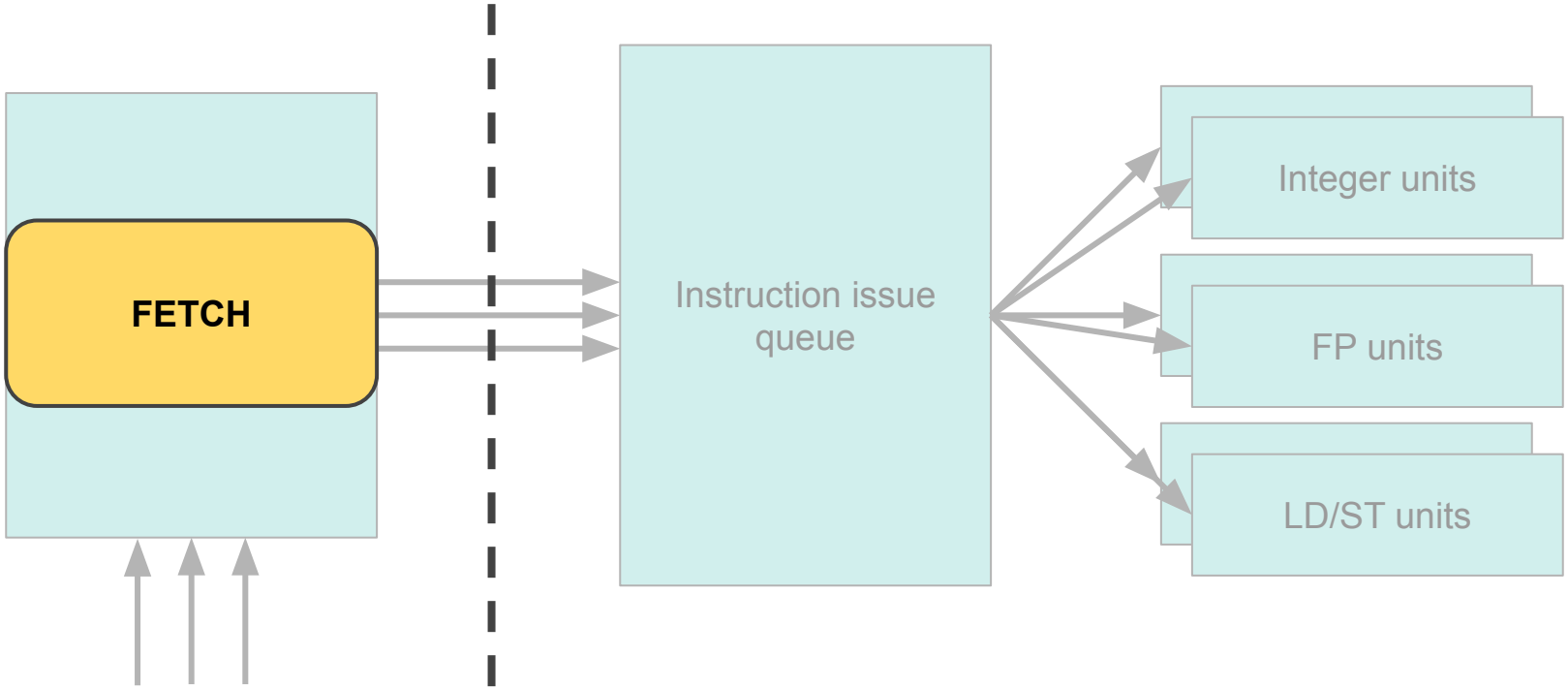
A **3-issue processor** issues 3 instructions per cycle

Motivation

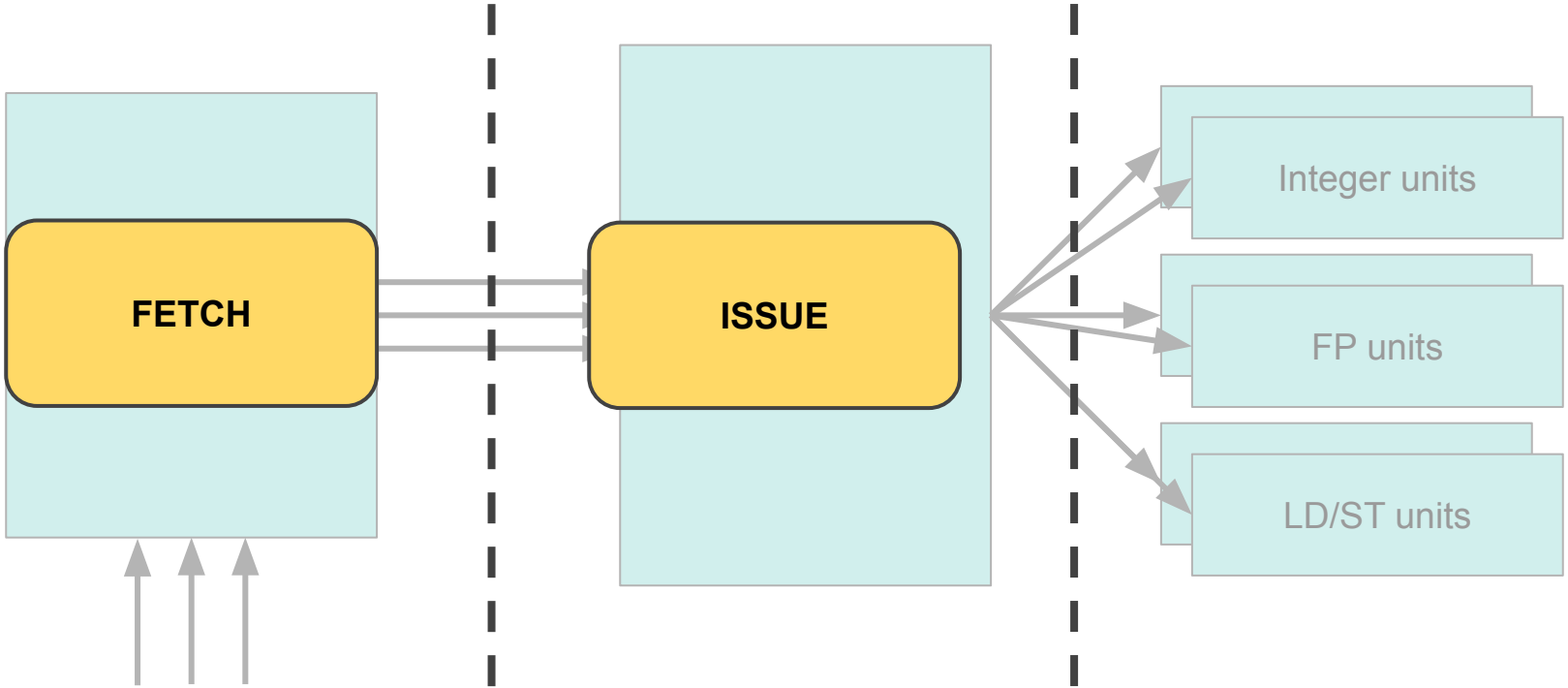
Is this scalable?



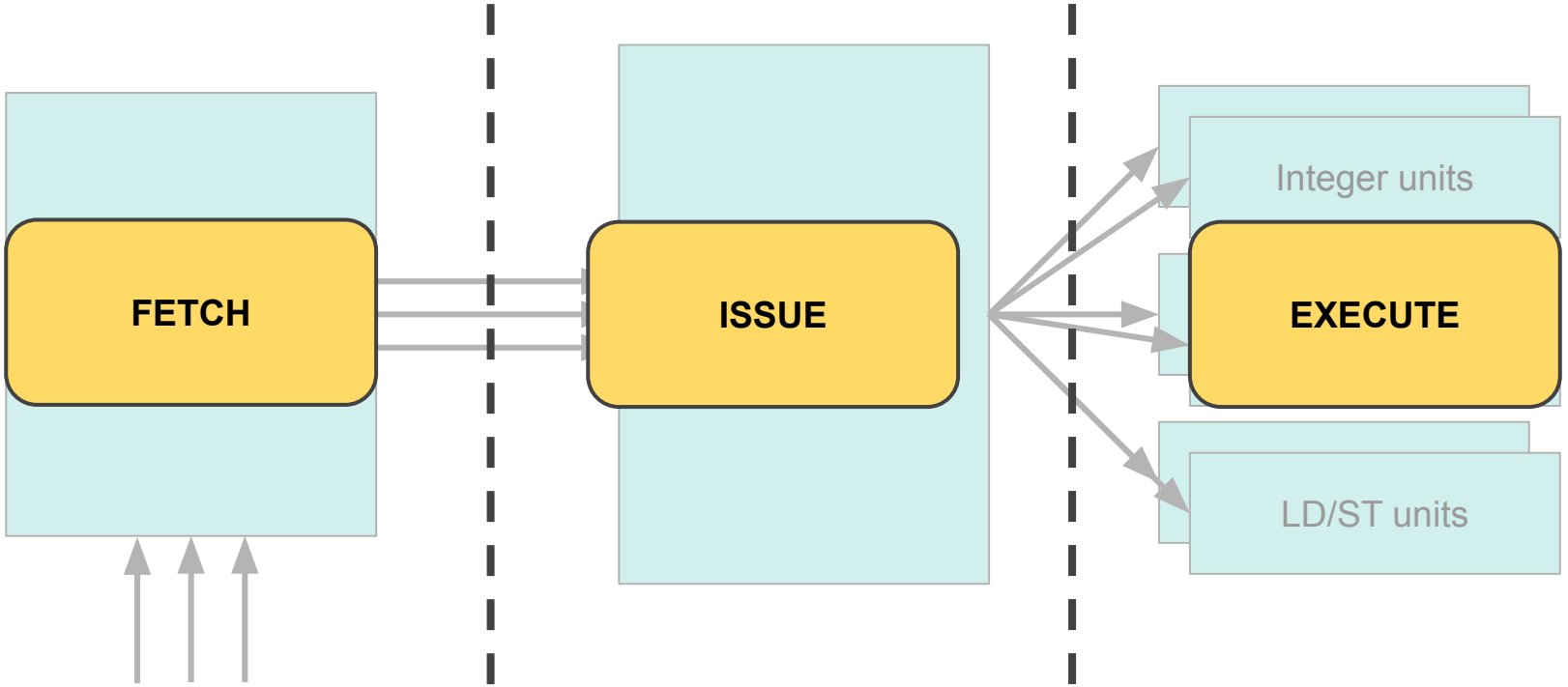
Is this scalable?



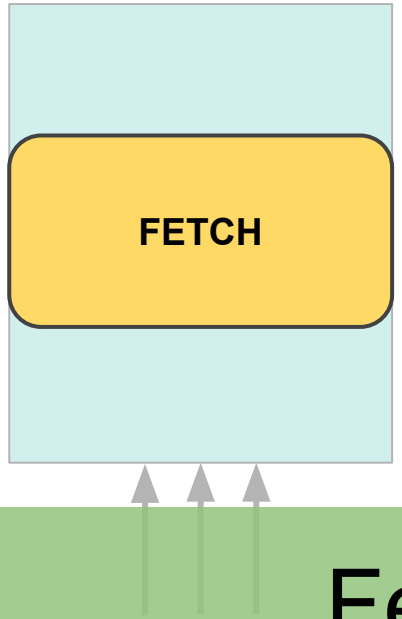
Is this scalable?



Is this scalable?



Scaling fetch phase

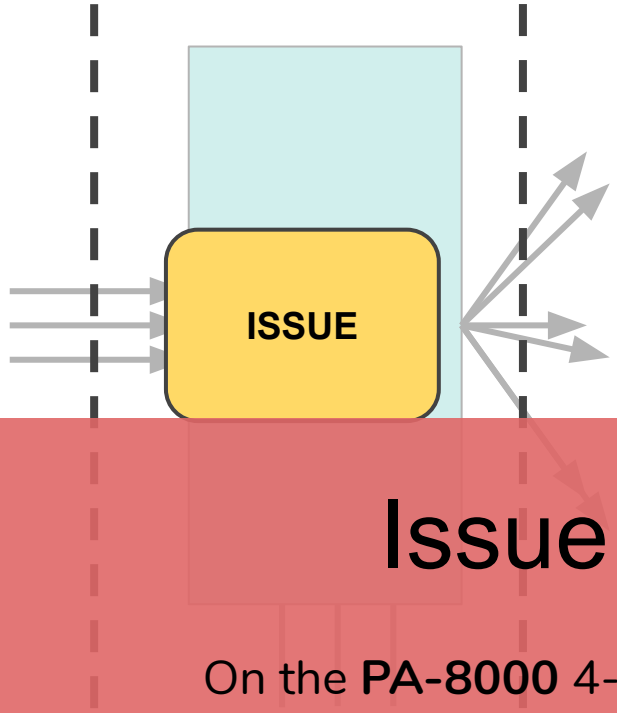


- Requires **fast instruction cache** with **good hit rate**
- Requires a **good branch predictor**

- With little space we are able to have a **misprediction rate under 5%** for most programs
- Cache misses are hidden by the size of the issue queue

Fetch phase will scale

Scaling issue phase



- Requires a bigger **issue queue**
- Requires more **instruction renaming** logic

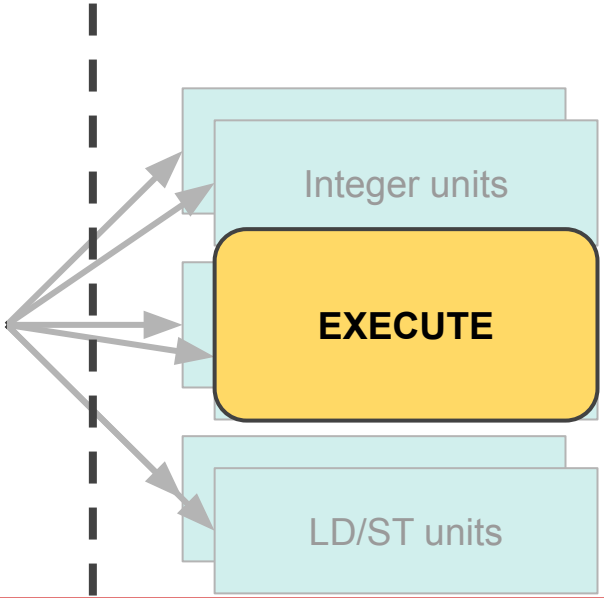
- Issue queue **grows quadratically** with the issue width
- The comparaisons in the renaming logic also grow quadratically
- Wires get longer, and therefore also cycle time

Issue phase will **not** scale

On the **PA-8000** 4-issue, 56 instructions queue: 20% of the die area

add R8, R17, R18

Scaling execute phase

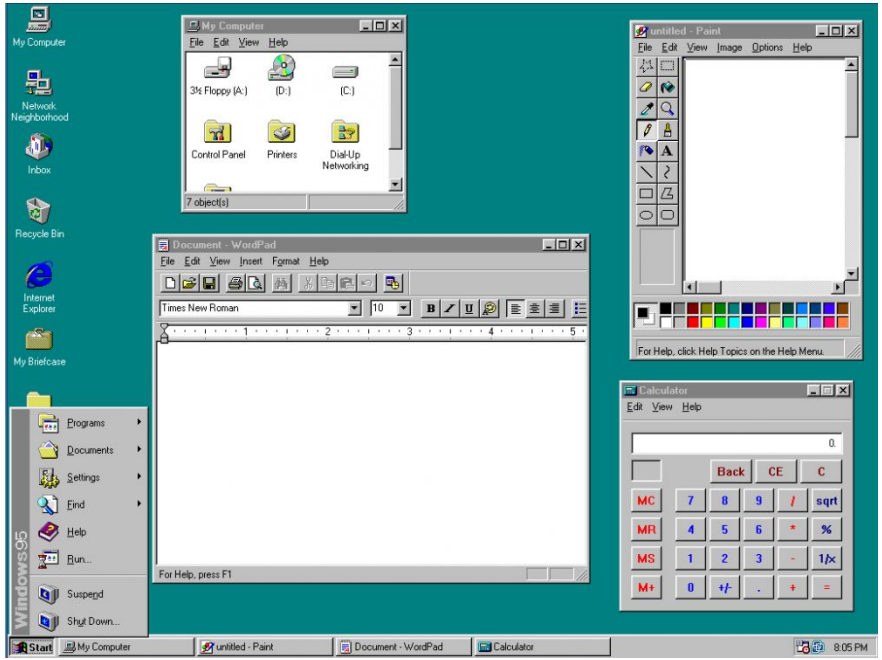


- Requires more execution units
- Requires more data cache ports
- Requires more register file ports

- Execution units **grows linearly**
- The complexity of the register file **grows quadratically**
- **Longer delays** in the data cache
- **Longer wires** in the bypass logic

Execute phase will **not** scale

Other motivation: Applications

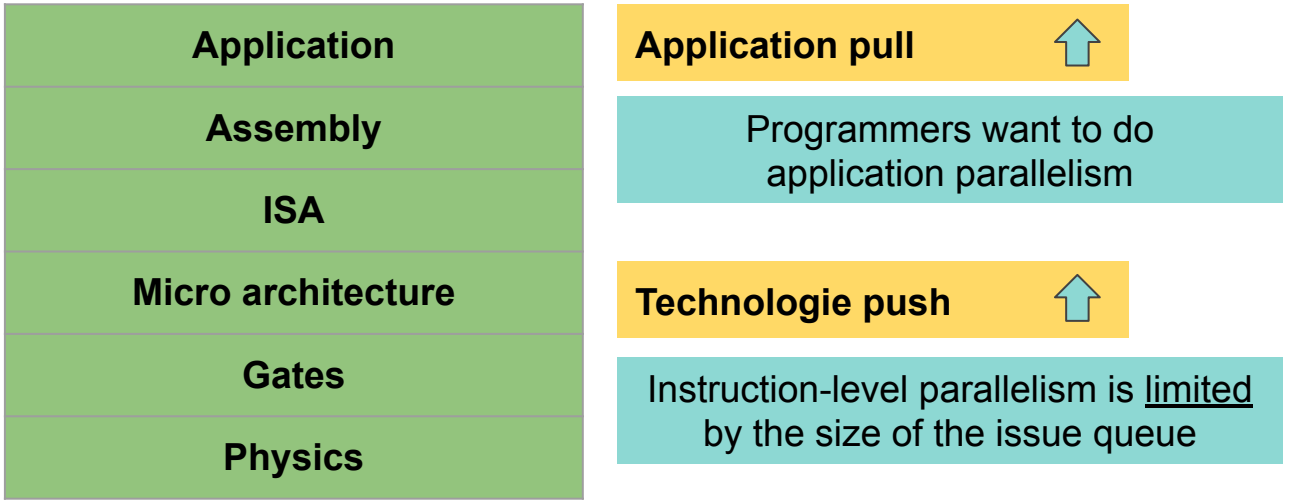


- Windows 95 is mono-processor
- Windows 96 is **multi-processor**

- Programmers want to do **multithreading** in their app

- **Automatic parallelization** technology emerges

Technology push and application pull



The case for a Single-chip Multiprocessor

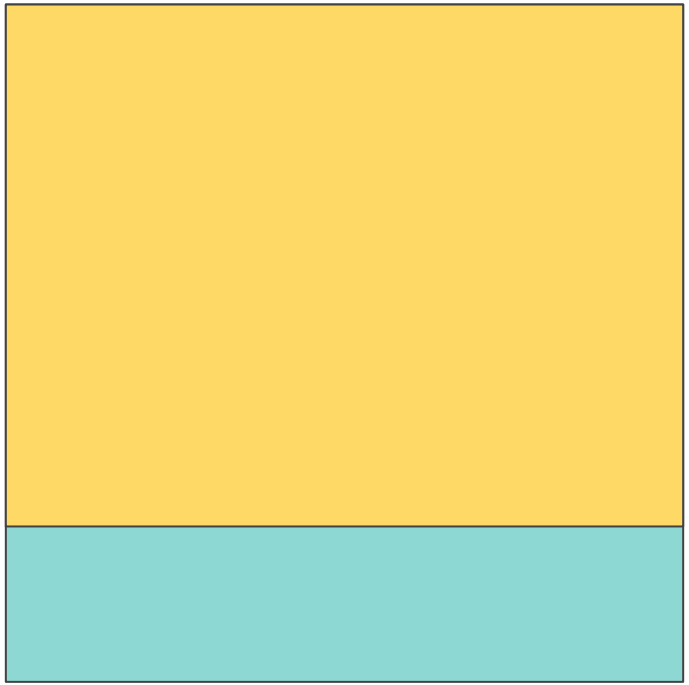
Comparing two Microarchitectures

21mm x 21mm

21mm x 21mm

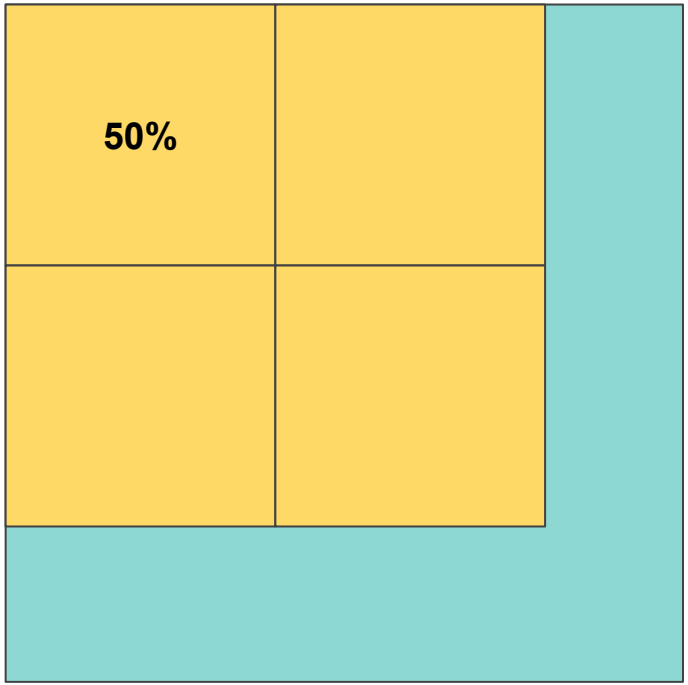
One CPU

21mm x 21mm



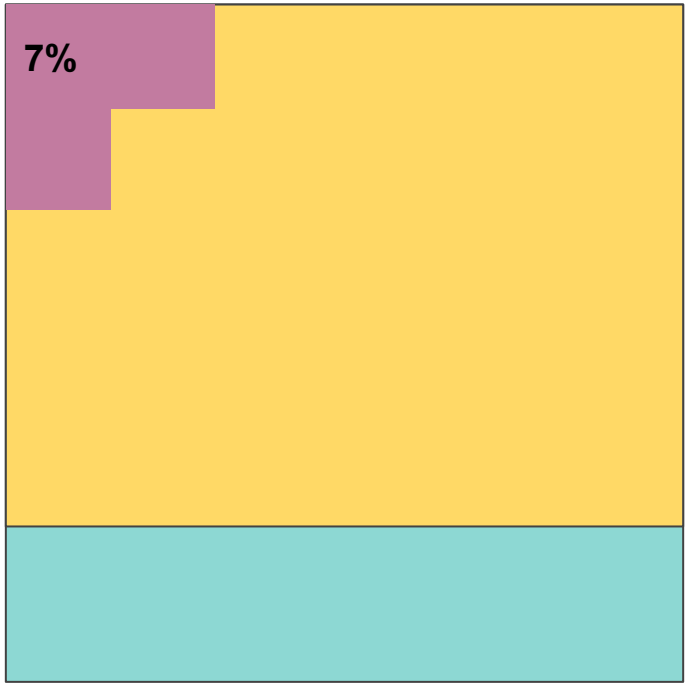
Four CPUs

21mm x 21mm



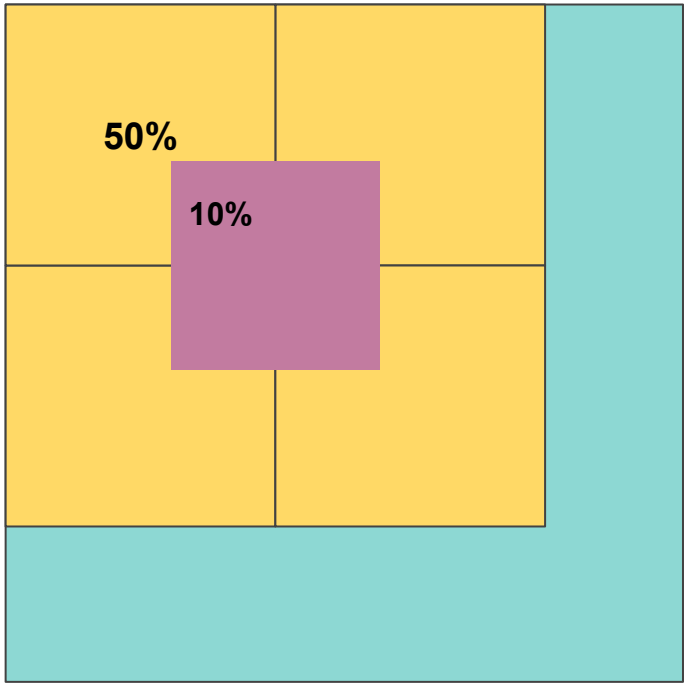
3 integer units

21mm x 21mm



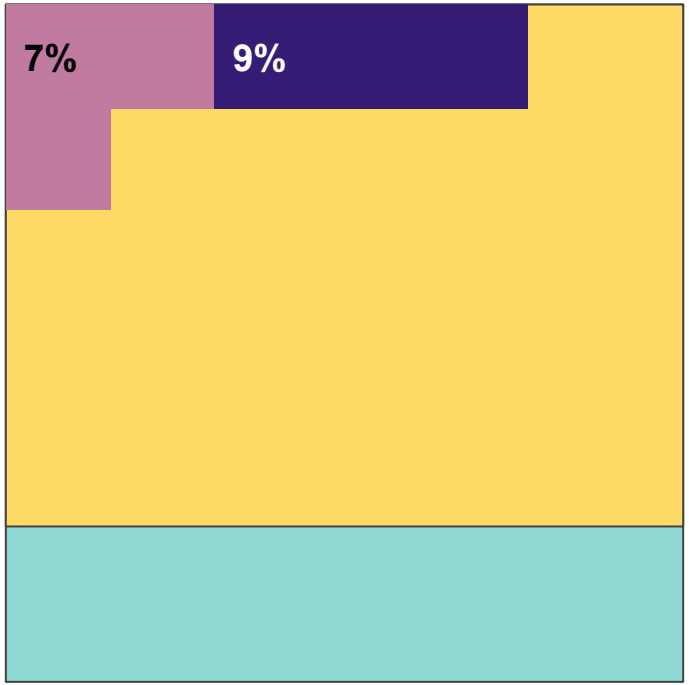
4x1 integer units

21mm x 21mm



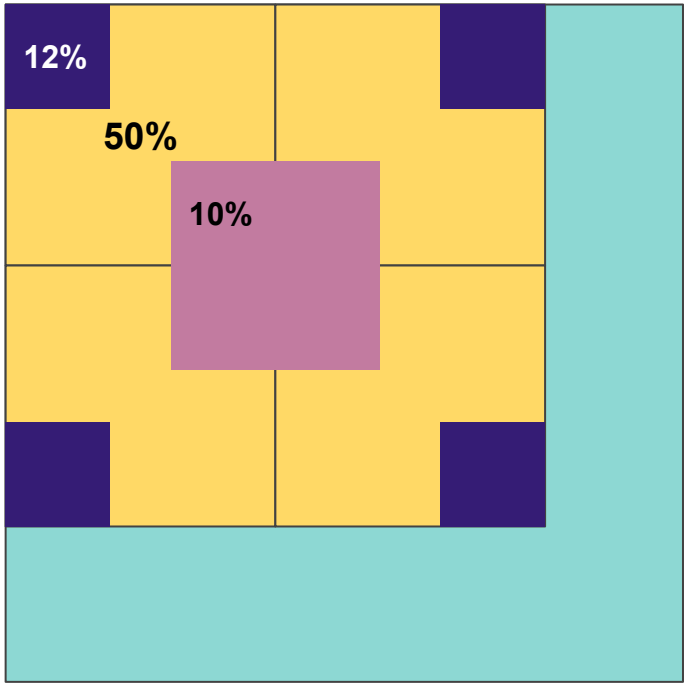
3 FP units

21mm x 21mm



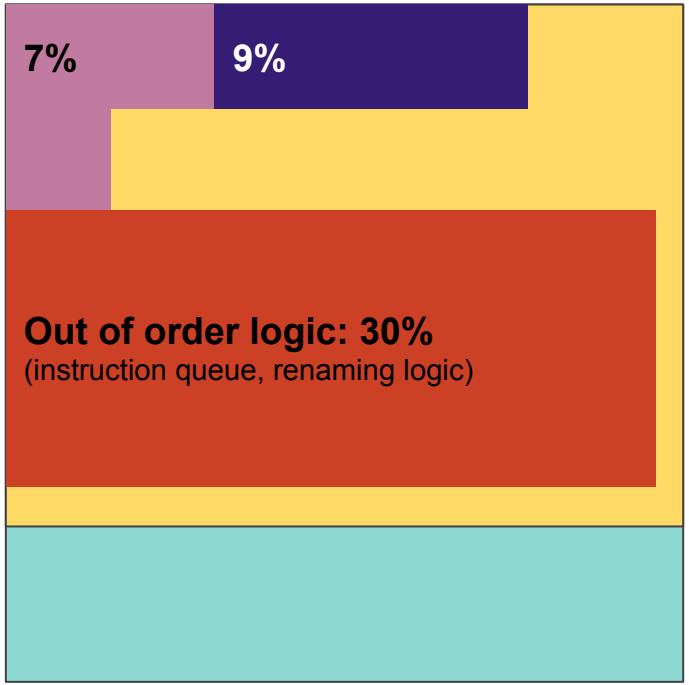
4x1 FP units

21mm x 21mm



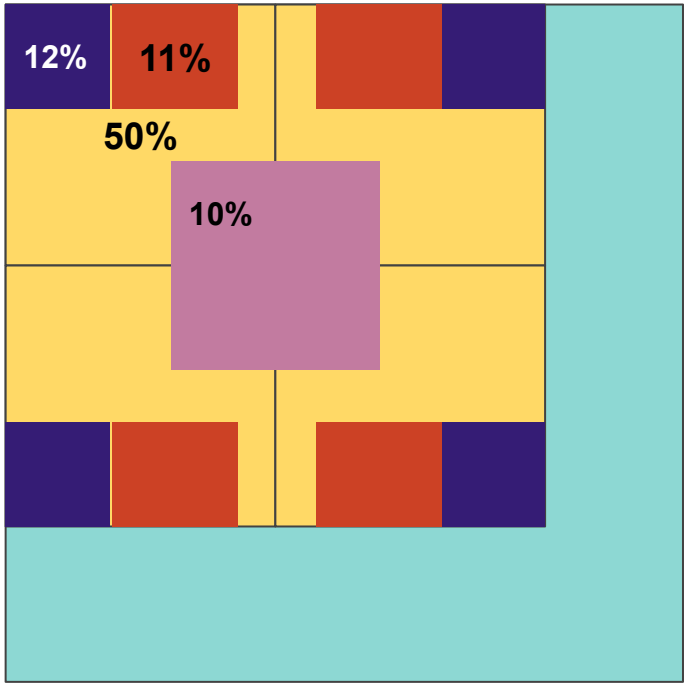
Issue width: 6
instruction queue: 128 entries

21mm x 21mm



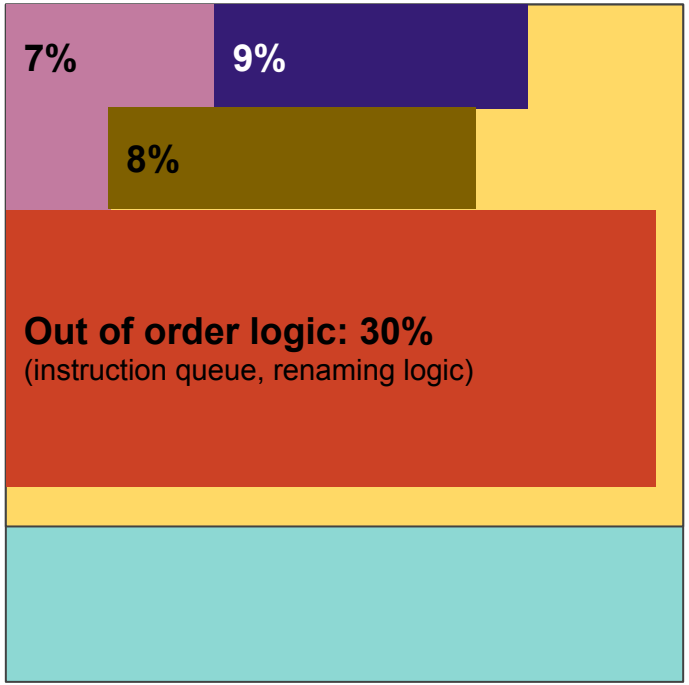
Issue width: 4x2
instruction queues: 4x8 entries

21mm x 21mm



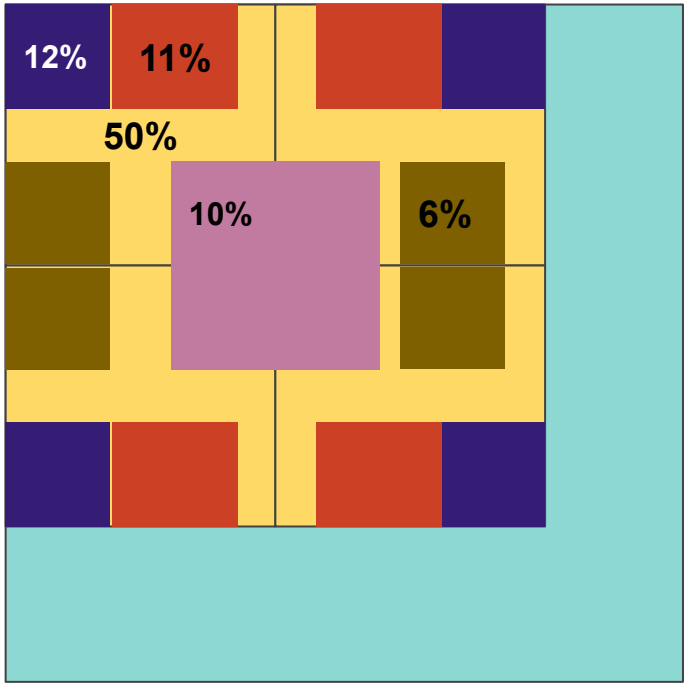
L1 cache
Hit time: 2 cycles
8 Banks

21mm x 21mm



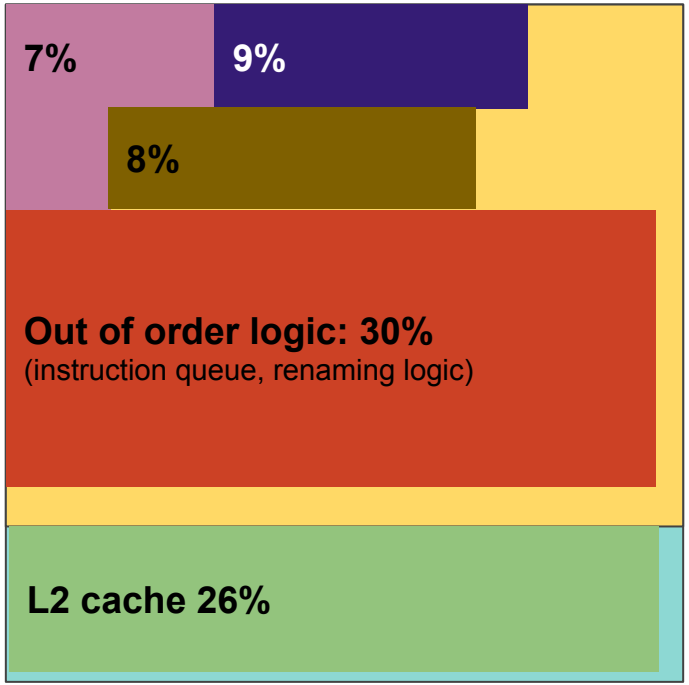
L1 cache
Hit time: 1 cycle

21mm x 21mm



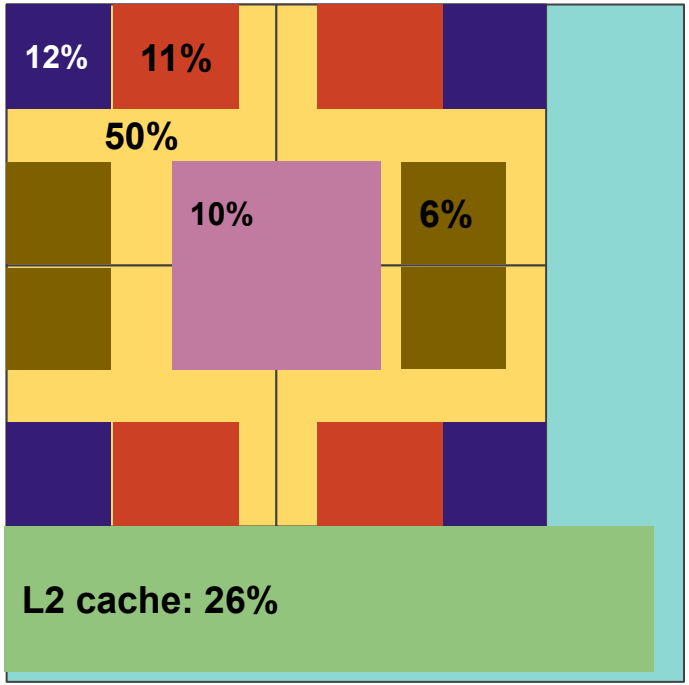
L2 Cache:
Size: 256K
Hit time: 4 cycles

21mm x 21mm



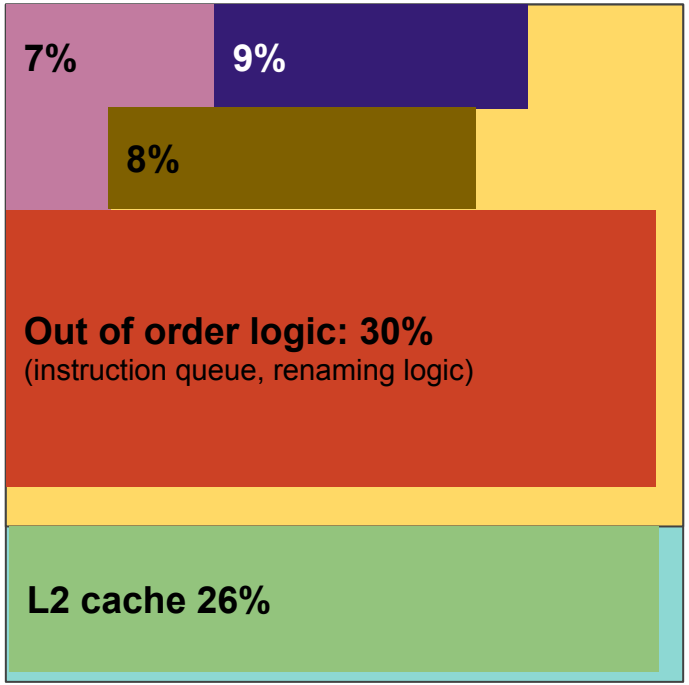
L2 cache:
Size: 256K
Hit time: 5 cycles

21mm x 21mm



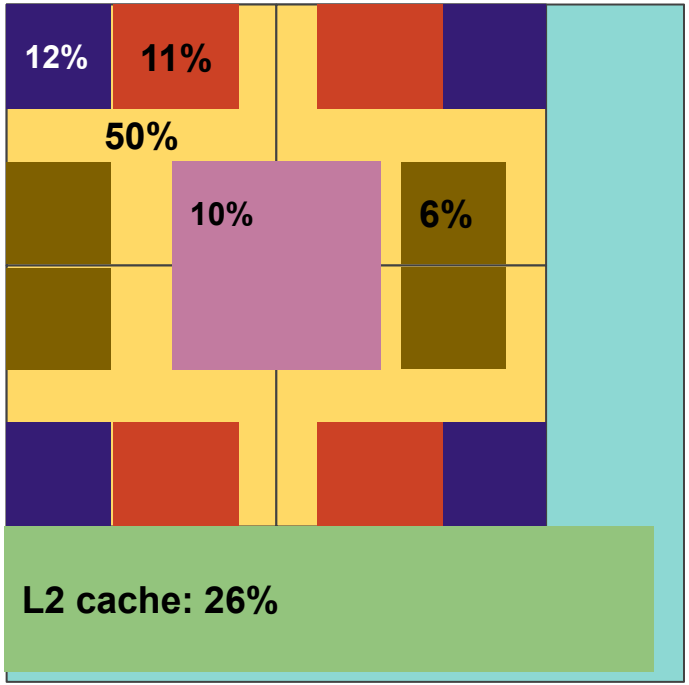
Clock: 500 MHz

21mm x 21mm



Clock: 500 MHz

21mm x 21mm



Simulation

Tested applications

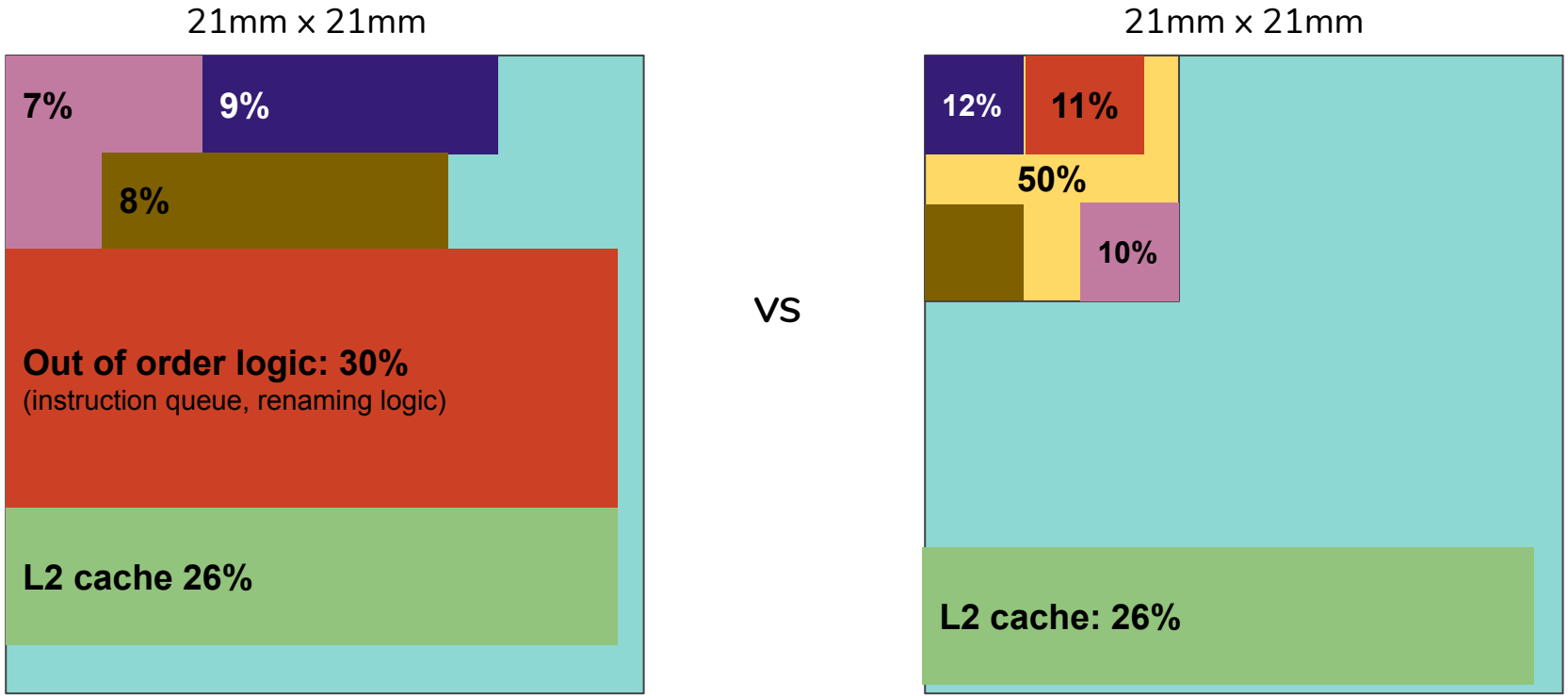
	Integer applica
compress	compresses and uncompresses file in m
eqntott	translates logic equations into truth tabl
m88ksim	Motorola 88000 CPU simulator
MPsim	VCS compiled Verilog simulation of a r
	Floating point appl
applu	solver for parabolic/elliptic partial differ
apsi	solves problems of temperature, wind, v
swim	shallow water model with 1K × 1K grid
tomcatv	mesh-generation with Thompson solver
	Multiprogramming application
pmake	parallel make of gnuchess using C compiler

Table 4. The applications.

We use SimOS because it supports multiprocessor

Programs are manually edited to be multithreaded

6-issues vs 2-issues



Simulation: 6-issues vs 2-issues

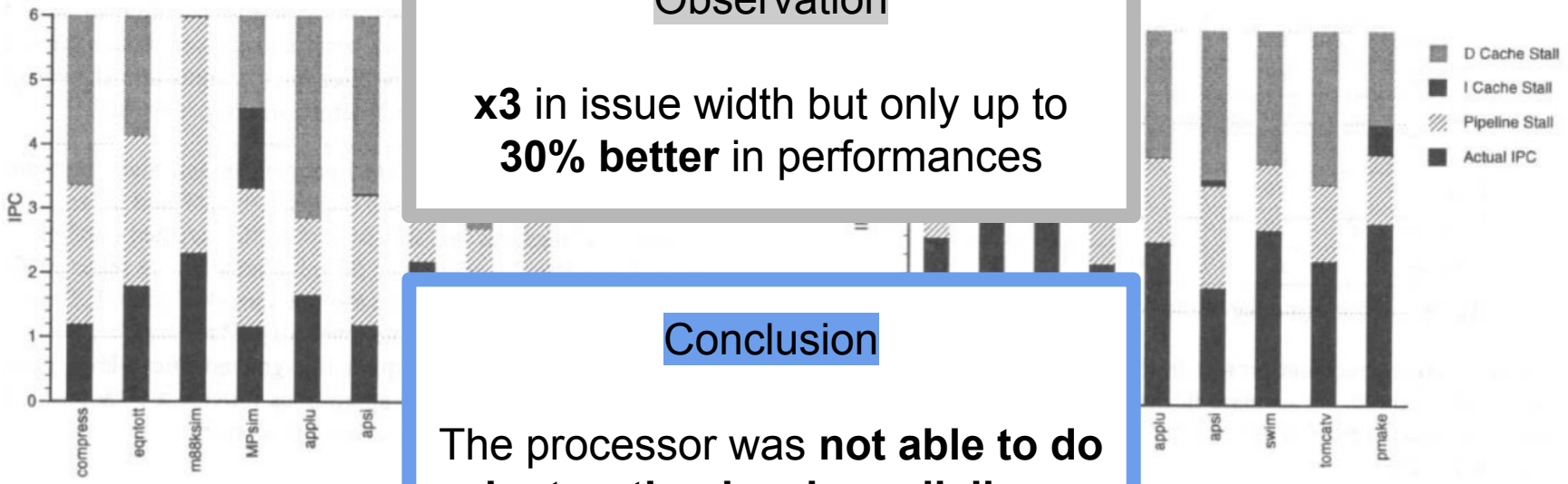


Figure 5. IPC Breakdown for **own for a single 2-issue processor.**

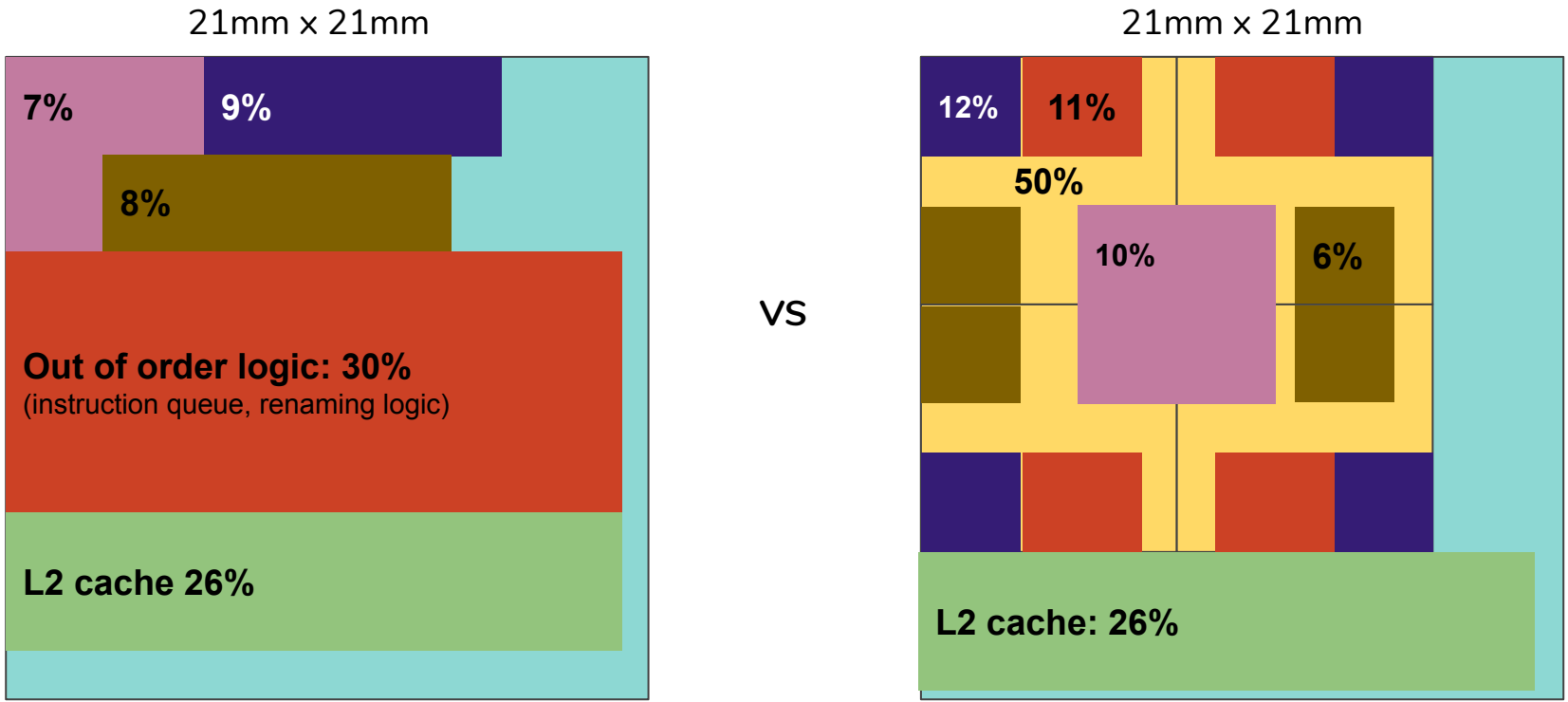
Observation

x3 in issue width but only up to **30% better** in performances

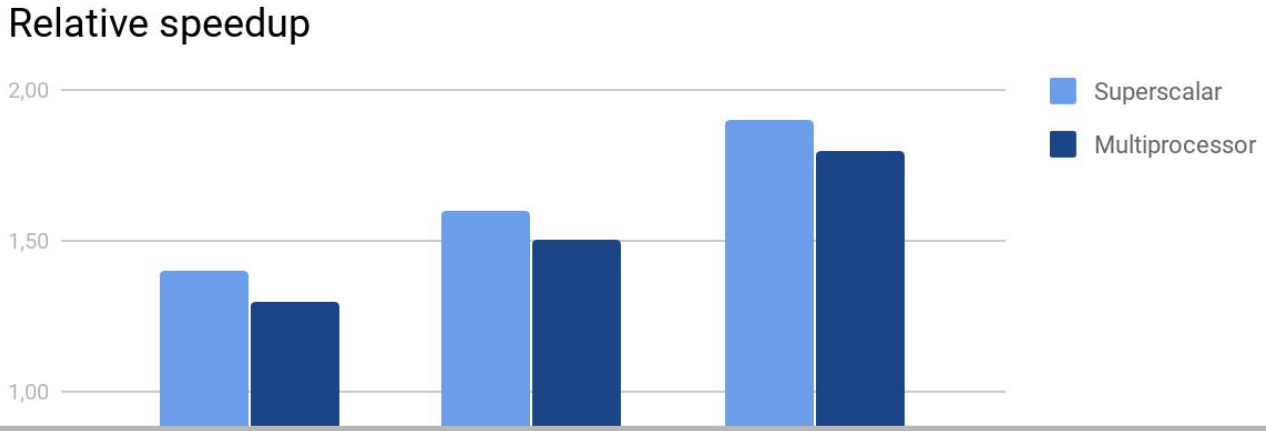
Conclusion

The processor was **not able to do instruction level parallelism**

Simulation: 6-issues vs 4x2-issues



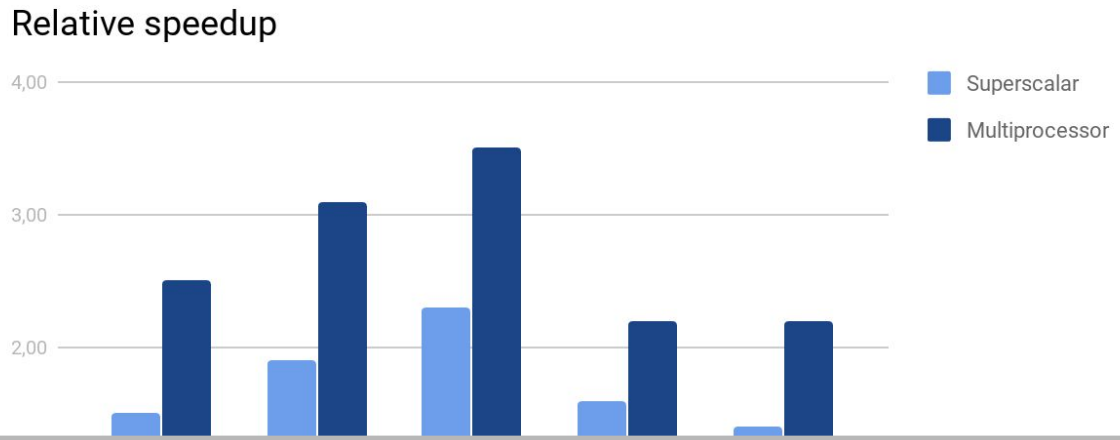
Simulation: 6-issues vs 4x2-issues



Observation

Fine grained thread-level parallelism: Superscalar does at most **10% better**

Simulation: 6-issues vs 4x2-issues



Observation

Large grained thread-level parallelism: Multiprocessor performs **50% - 100% better**

Conclusion

Conclusion

- Background: The size reduction of transistors gives us opportunities to innovate
- Problem: Increasing the number of issues in superscalar architectures is not sustainable
- Goal: Design simpler processors with multiple smaller CPUs
- Key contribution:
  - **Demonstration:** Prove that superscalar architectures are not scalable
  - **Innovation:** Design and compare a superscalar architecture and a multiprocessor architecture
  - **Interpretation:** Identify different types of applications and compare their performances on each architecture

Seminar in computer architecture

The Case for a Single-Chip Multiprocessor

Kunle Olukotun, Basem A. Nayfeh, Lance Hammond, Ken Wilson, and Kunyung Chang

ASPLOS 1996

Presented by Gauthier de Chezelles

Discussion

Strengths

- The paper is trying to project itself on the long term
- The architectural choices are well argumented (especially on latency and queue sizes)
- The results are well analyzed

Weaknesses

- Does not speak about the need of recompiling
- Not a lot of details on how well are the apps threaded.
- Comparaison with a perfectly multithreaded app would have been nice
- Not a lot of background (register renaming, register file....). But maybe it was different times
- Only compares superscalar and multiprocessor. Could we have done something else with those new transistors ?
- No mention of energy

Open discussion

- What would you have done with those new transistors?
  - What about more instructions ? Vector instructions ?
- What about today ? What would you do with a new transistor scaling ?
  - Do you see any limit in having 16, 64 CPUs in one processor ?
  - If yes what other use cases could you think of ?

Annex

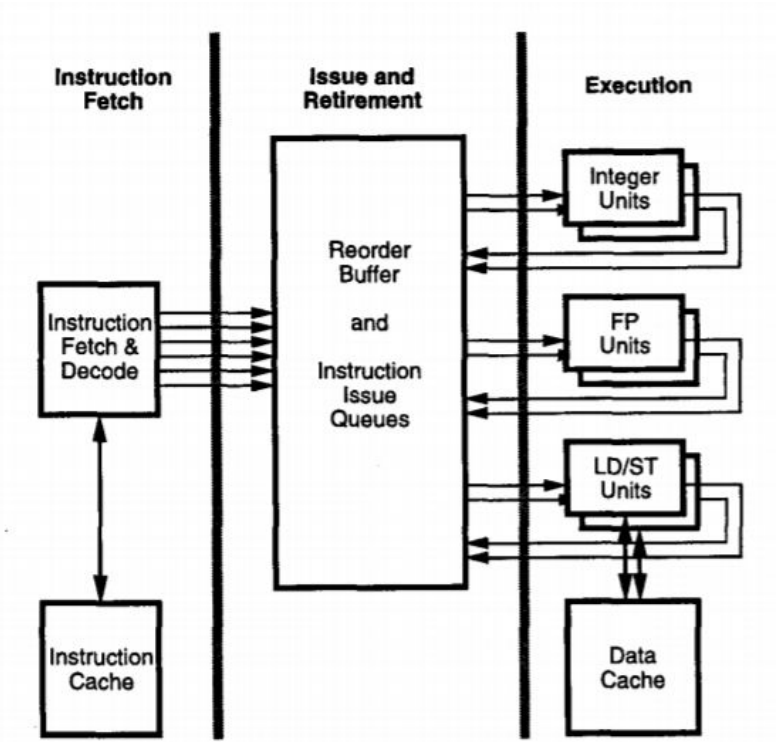


Figure 1. A dynamic superscalar CPU

Annex

	6-way SS	4x2-way MP
# of CPUs	1	4
Degree superscalar	6	4 x 2
# of architectural registers	32int / 32fp	4 x 32int / 32fp
# of physical registers	160int / 160fp	4 x 40int / 40fp
# of integer functional units	3	4 x 1
# of floating pt. functional units	3	4 x 1
# of load/store ports	8 (one per bank)	4 x 1
BTB size	2048 entries	4 x 512 entries
Return stack size	32 entries	4 x 8 entries
Instruction issue queue size	128 entries	4 x 8 entries
I cache	32 KB, 2-way S. A.	4 x 8 KB, 2-way S. A.
D cache	32 KB, 2-way S. A.	4 x 8 KB, 2-way S. A.
L1 hit time	2 cycles (4 ns)	1 cycle (2 ns)
L1 cache interleaving	8 banks	N/A
Unified L2 cache	256 KB, 2-way S. A.	256 KB, 2-way S. A.
L2 hit time / L1 penalty	4 cycles (8 ns)	5 cycles (10 ns)
Memory latency / L2 penalty	50 cycles (100 ns)	50 cycles (100 ns)

Table 1. Key characteristics of the two microarchitectures

Annex

CPU Component	0.35µm R10K Original Size (mm^2)	Size Extrapolated to 0.25µm (mm^2)	% Growth Due to New Functionality	New Size (mm^2)	% Area
256K On-Chip L2 Cache [a]	219	112	0%	112	26%
8-bank D Cache (32 KB)	26	13	25%	17	4%
8-bank I Cache (32 KB)	28	14	25%	18	4%
TLB Mechanism	10	5	200%	15	3%
External Interface Unit	27	14	0%	14	3%
Instruction Fetch Unit and BTB	18	9	200%	28	6%
Instruction Decode Section	21	11	250%	38	9%
Instruction Queues	28	14	250%	50	12%
Reorder Buffer	17	9	300%	34	9%
Integer Functional Units	20	10	200%	31	7%
FP Functional Units	24	12	200%	37	9%
Clocking & Overhead	73	37	0%	37	9%
Total Size	—	—	—	430	100%

Table 2. Size extrapolations for the 6-way superscalar from the MIPS R10000 processor

Annex

CPU Component	0.35µm R10K Original Size (mm^2)	Size Extrapolated to 0.25µm (mm^2)	% Growth Due to New Functionality	New Size (mm^2)	% Area (of CPU / of entire chip)
D Cache (8 KB)	26	13	-75%	3	6% /3%
I Cache (8 KB)	28	14	-75%	4	7% / 3%
TLB Mechanism	10	5	0%	5	9% / 5%
Instruction Fetch Unit and BTB	18	9	-25%	7	13% / 7%
Instruction Decode Section	21	11	-50%	5	10% / 5%
Instruction Queues	28	14	-70%	4	8% / 4%
Reorder Buffer	17	9	-80%	2	3% / 2%
Integer Functional Units	20	10	0%	10	20% / 10%
FP Functional Units	24	12	0%	12	23% / 12%
Per-CPU Subtotal	—	—	—	53	100% / 50%
256K On-Chip L2 Cache [a]	219	112	0%	112	26%
External Interface Unit	27	14	0%	14	3%
Crossbar Between CPUs	—	—	—	50	12%
Clocking & Overhead	73	37	0%	37	9%
Total Size	—	—	—	424	100%

Table 3. Size extrapolations in the 4 × 2-way MP from the MIPS R10000 processor.

Annex

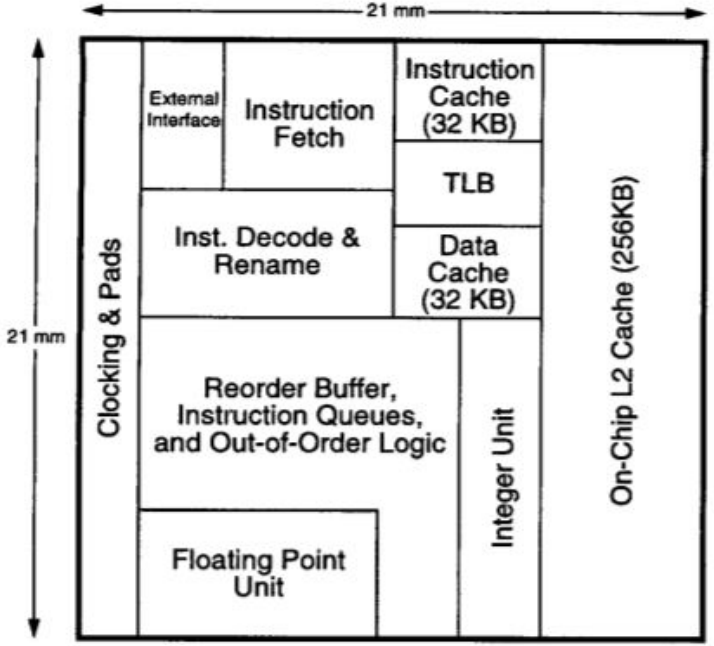


Figure 2. Floorplan for the six-issue dynamic superscalar microprocessor.

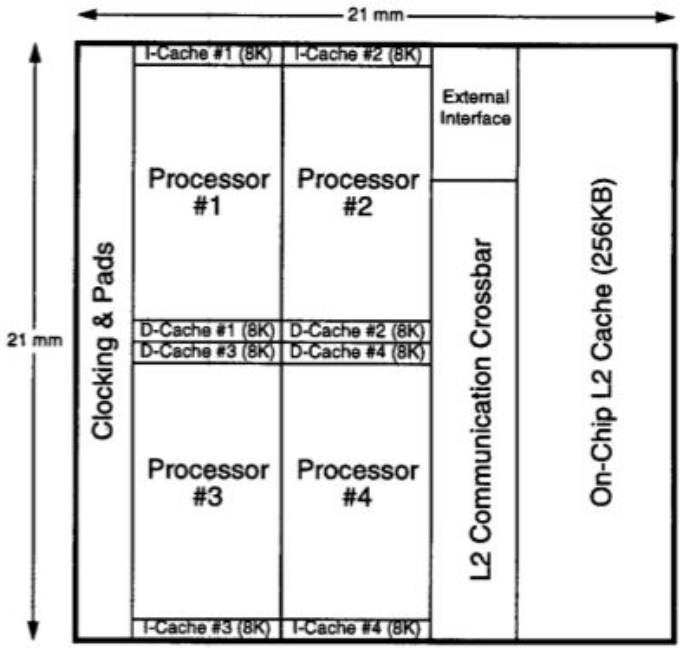


Figure 3. Floorplan for the four-way single-chip multiprocessor.