CVA6S+: A Superscalar RISC-V Core with High-Throughput Memory Architecture

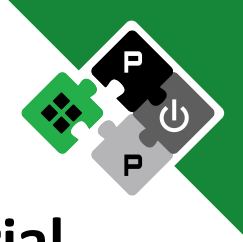
Riccardo Tedeschi*[1]*, *Gianmarco Ottavi*[1]*, Côme Allart*[2,3]*, Nils Wistoff*[4]*, Zexin Fu*[4]*, Filippo Grillotti*[5]*, Fabio De Ambroggi*[5]*, Elio Guidetti*[5]*, Jean-Baptiste Rigaud*[3]*, Olivier Potin*[3]*, Jean Roch Coulon*[2]*, César Fuguet*[6]*, Luca Benini*[1,4]*, Davide Rossi*[1]

University of Bologna, Italy[1]
Mines Saint-Etienne, France[3]
STMicroelectronics, Italy[5]
Thales DIS, France[2]
ETH Zürich, Switzerland[4]
Inria, France[6]

PULP Platform
Open Source Hardware, the way it should be!

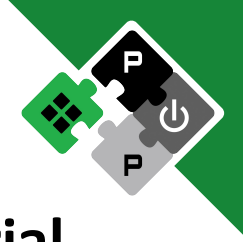
@pulp_platform
pulp-platform.org
youtube.com/pulp_platform

Introduction: RISC-V high-performance open cores



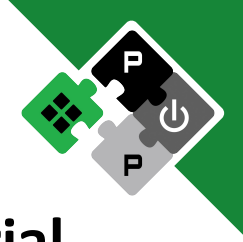
➔ The **growing demand for autonomy** in critical fields like **automotive**, **industrial automation**, and **aerospace** is driving the need for **high-performance CPUs**

Introduction: RISC-V high-performance open cores



- ➔ The **growing demand for autonomy** in critical fields like **automotive**, **industrial automation**, and **aerospace** is driving the need for **high-performance CPUs**
- ➔ An increasing number of **open-source RISC-V cores** are targeting **high-performance applications**, including **BlackParrot**, **BOOM**, **XiangShan**, and **Xuantie C910**

Introduction: RISC-V high-performance open cores



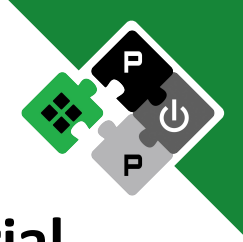
- ➔ The **growing demand for autonomy** in critical fields like **automotive**, **industrial automation**, and **aerospace** is driving the need for **high-performance CPUs**
- ➔ An increasing number of **open-source RISC-V cores** are targeting **high-performance applications**, including **BlackParrot**, **BOOM**, **XiangShan**, and **Xuantie C910**

CVA6[1] is a configurable **64/32 bit RISC-V core** originally developed by the **PULP Platform** and now maintained by **OpenHW Group** with the support of **multiple industrial and academic partners**:

[1]F. Zaruba, "The Cost of Application-Class Processing: Energy and Performance Analysis of a Linux-Ready 1.7-GHz 64-Bit RISC-V Core in 22-nm FDSOI Technology", IEEE VLSI, 2019

ETH zürich · ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA · MINES Saint-Étienne · Institut Mines-Télécom · Inria · STMicroelectronics · THALES

Introduction: RISC-V high-performance open cores



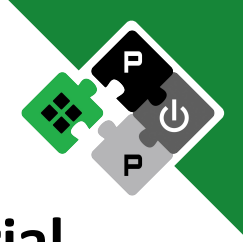
- ➔ The **growing demand for autonomy** in critical fields like **automotive**, **industrial automation**, and **aerospace** is driving the need for **high-performance CPUs**
- ➔ An increasing number of **open-source RISC-V cores** are targeting **high-performance applications**, including **BlackParrot**, **BOOM**, **XiangShan**, and **Xuantie C910**

CVA6[1] is a configurable **64/32 bit RISC-V core** originally developed by the **PULP Platform** and now maintained by **OpenHW Group** with the support of **multiple industrial and academic partners**:

- ➔ **6-stage** pipeline: two-stage Instruction Fetch (IF), Instruction Decode (ID), Instruction Issue (IS), Instruction Execute (IE), and Writeback (WB)

[1]F. Zaruba, "The Cost of Application-Class Processing: Energy and Performance Analysis of a Linux-Ready 1.7-GHz 64-Bit RISC-V Core in 22-nm FDSOI Technology", IEEE VLSI, 2019

Introduction: RISC-V high-performance open cores



- ➔ The **growing demand for autonomy** in critical fields like **automotive**, **industrial automation**, and **aerospace** is driving the need for **high-performance CPUs**
- ➔ An increasing number of **open-source RISC-V cores** are targeting **high-performance applications**, including **BlackParrot**, **BOOM**, **XiangShan**, and **Xuantie C910**

CVA6[1] is a configurable **64/32 bit RISC-V core** originally developed by the **PULP Platform** and now maintained by **OpenHW Group** with the support of **multiple industrial and academic partners**:

- ➔ **6-stage** pipeline: two-stage Instruction Fetch (IF), Instruction Decode (ID), Instruction Issue (IS), Instruction Execute (IE), and Writeback (WB)
- ➔ **In order dispatch**, **out of order completion, in order commit**

[1]F. Zaruba, "The Cost of Application-Class Processing: Energy and Performance Analysis of a Linux-Ready 1.7-GHz 64-Bit RISC-V Core in 22-nm FDSOI Technology", IEEE VLSI, 2019

Background: from CVA6 to CVA6S

CVA6 **IPC (Instructions Per Clock)** is **constrained by its simple, scalar in-order front-end** microarchitecture

[2]C. Allart, "Using a Performance Model to Implement a Superscalar CVA6", ACM CF'24

Background: from CVA6 to CVA6S

CVA6 **IPC (Instructions Per Clock)** is **constrained by its simple, scalar in-order front-end** microarchitecture

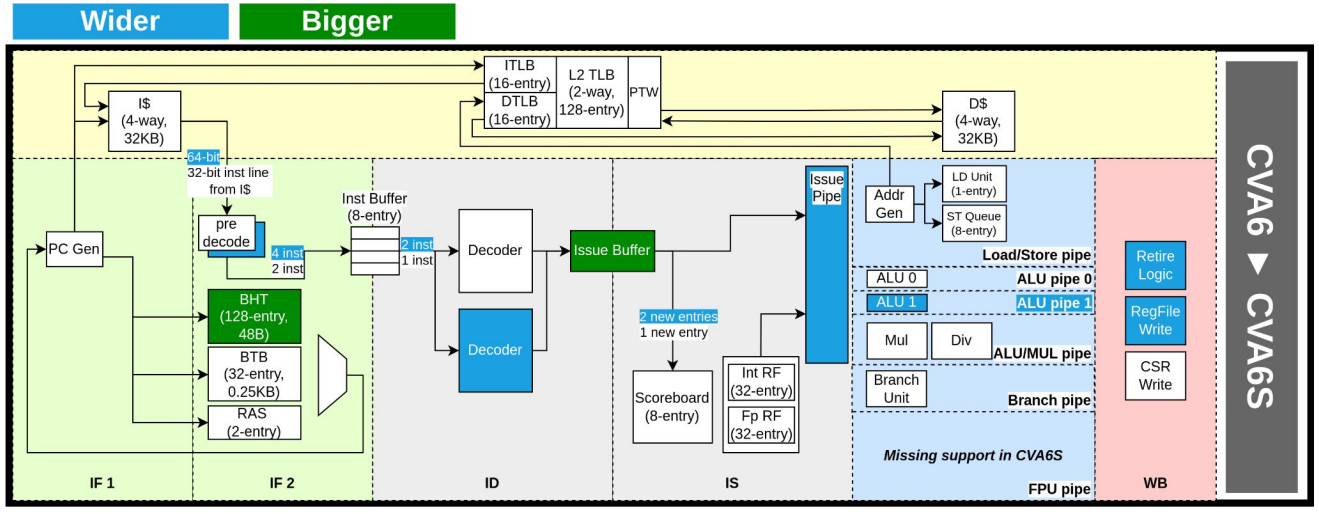
CVA6S[2] is the **<u>superscalar dual-issue version</u> of CVA6** by **Thales**, making the core suitable for **more demanding workloads**

[2]C. Allart, "Using a Performance Model to Implement a Superscalar CVA6", ACM CF'24

Background: from CVA6 to CVA6S

CVA6 **IPC (Instructions Per Clock)** is **constrained by its simple, scalar in-order front-end** microarchitecture

CVA6S[2] is the **superscalar dual-issue version of CVA6** by **Thales**, making the core suitable for **more demanding workloads**



- → ×2 instruction fetch width
- → ×2 decoding and issue logic
- → Secondary ALU

[2]C. Allart, "Using a Performance Model to Implement a Superscalar CVA6", ACM CF'24

Contribution: from CVA6S to CVA6S+

We present **CVA6S+**, which **builds on the CVA6S microarchitecture** with key enhancements aimed at **further boosting performance**:

Contribution: from CVA6S to CVA6S+

We present **CVA6S+**, which **builds on the CVA6S microarchitecture** with key enhancements aimed at **further boosting performance**:

➔ **Register renaming**

Contribution: from CVA6S to CVA6S+

We present **CVA6S+**, which **builds on the CVA6S microarchitecture** with key enhancements aimed at **further boosting performance**:

➔ **Register renaming**

➔ **Improved branch predictor**

Contribution: from CVA6S to CVA6S+

We present **CVA6S+**, which **builds on the CVA6S microarchitecture** with key enhancements aimed at **further boosting performance**:

- ➔ **Register renaming**
- ➔ **Improved branch predictor**
- ➔ **ALU-ALU forwarding**

Contribution: from CVA6S to CVA6S+

We present **CVA6S+**, which **builds on the CVA6S microarchitecture** with key enhancements aimed at **further boosting performance**:

- ➔ **Register renaming**
- ➔ **Improved branch predictor**
- ➔ **ALU-ALU forwarding**
- ➔ **FPU integration in superscalar mode**

Contribution: from CVA6S to CVA6S+

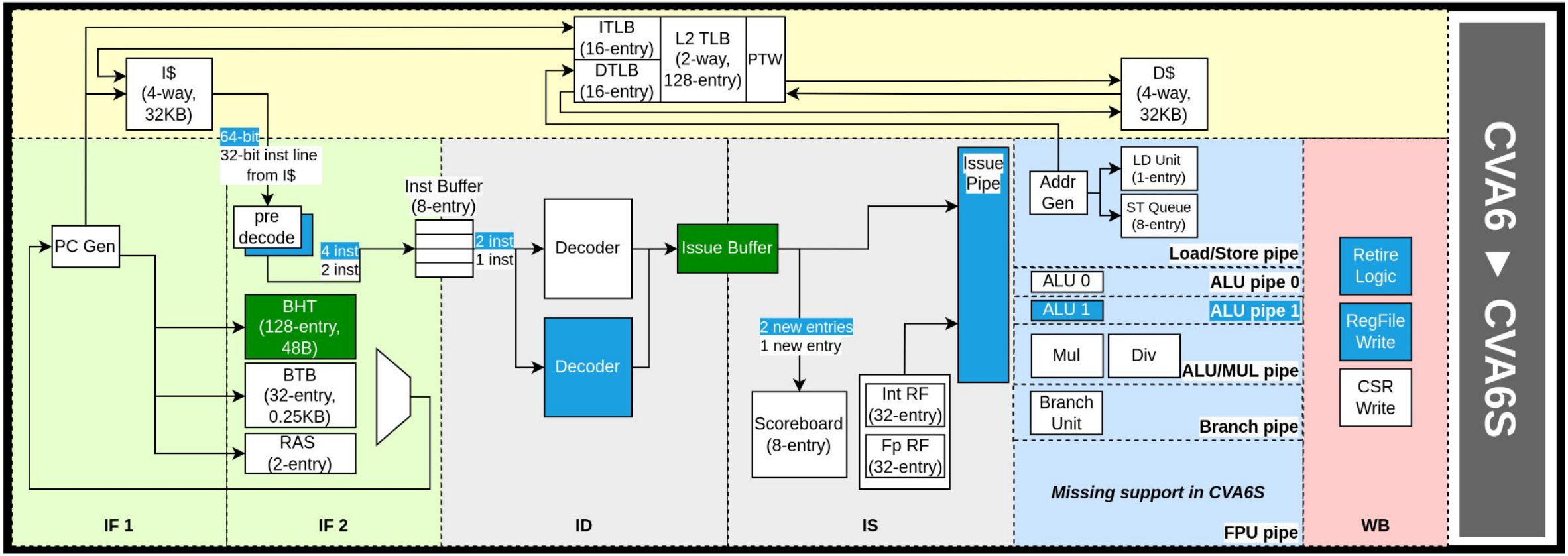
We present **CVA6S+**, which **builds on the CVA6S microarchitecture** with key enhancements aimed at **further boosting performance**:

- ➔ **Register renaming**
- ➔ **Improved branch predictor**
- ➔ **ALU-ALU forwarding**
- ➔ **FPU integration in superscalar mode**

Moreover, we integrate and evaluate **CVA6S+** with the the OpenHW Core-V High-Performance L1 Data Cache (**HPDCache**)

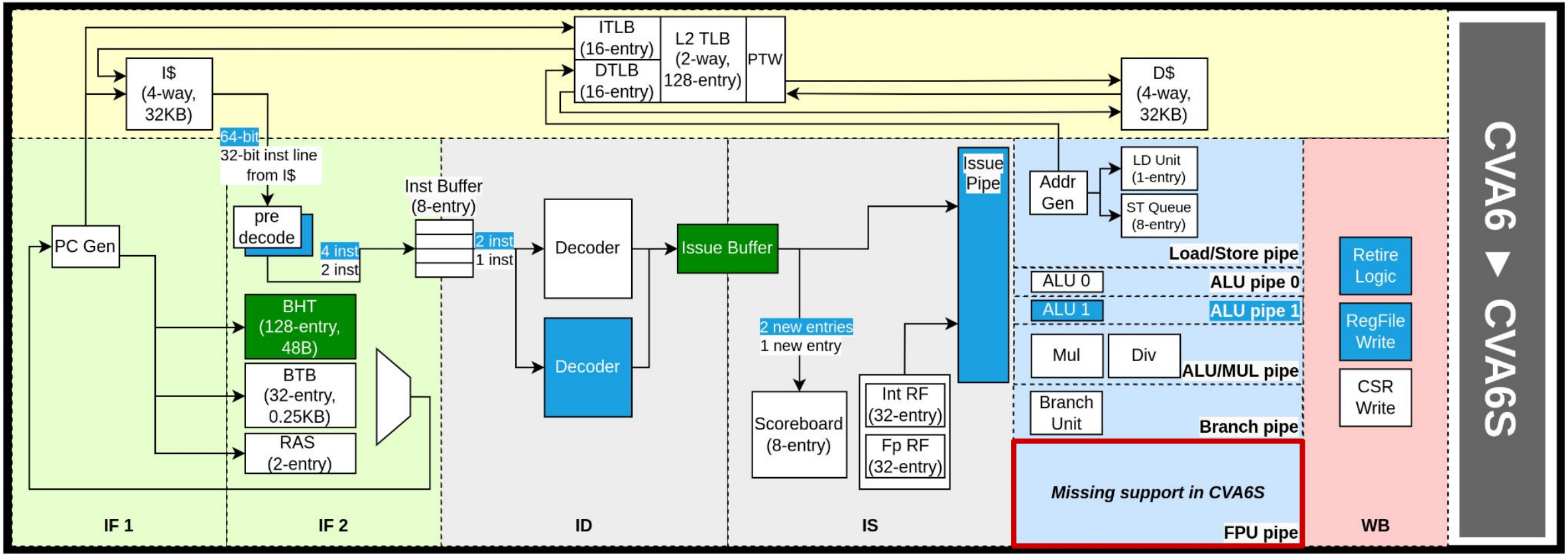
CVA6S: the baseline

Wider **Bigger**



CVA6S: the baseline

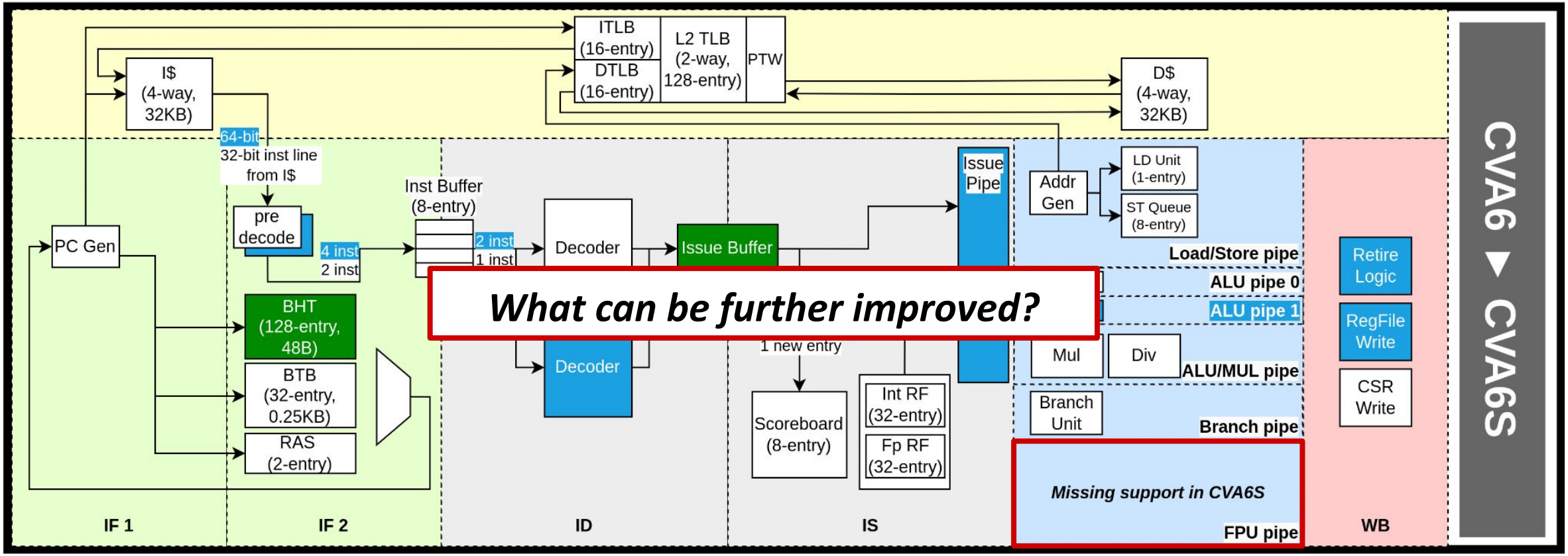
Wider **Bigger**



FPU support was **out of scope** for **CVA6S**

CVA6S: the baseline

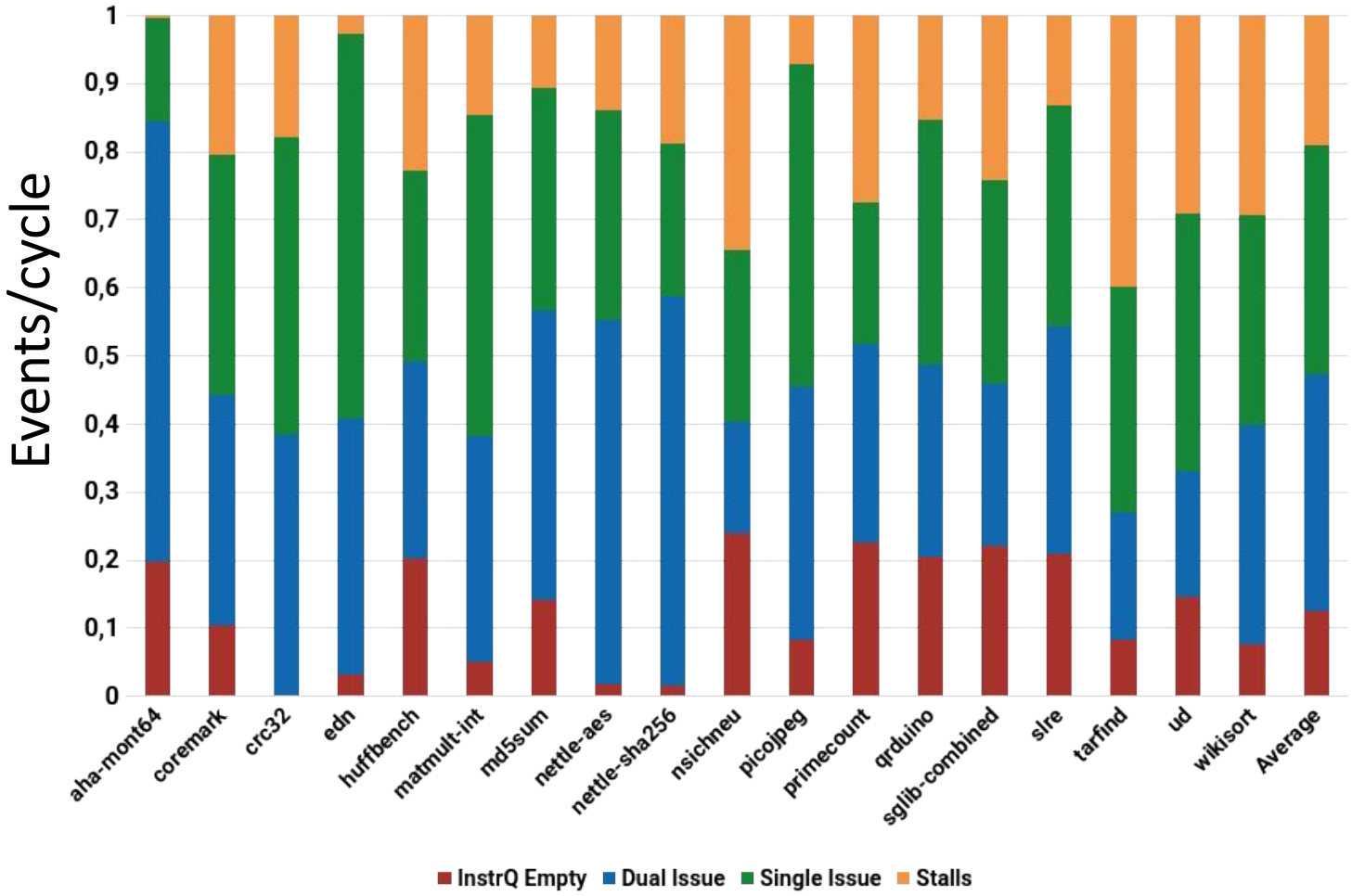
Wider **Bigger**



What can be further improved?

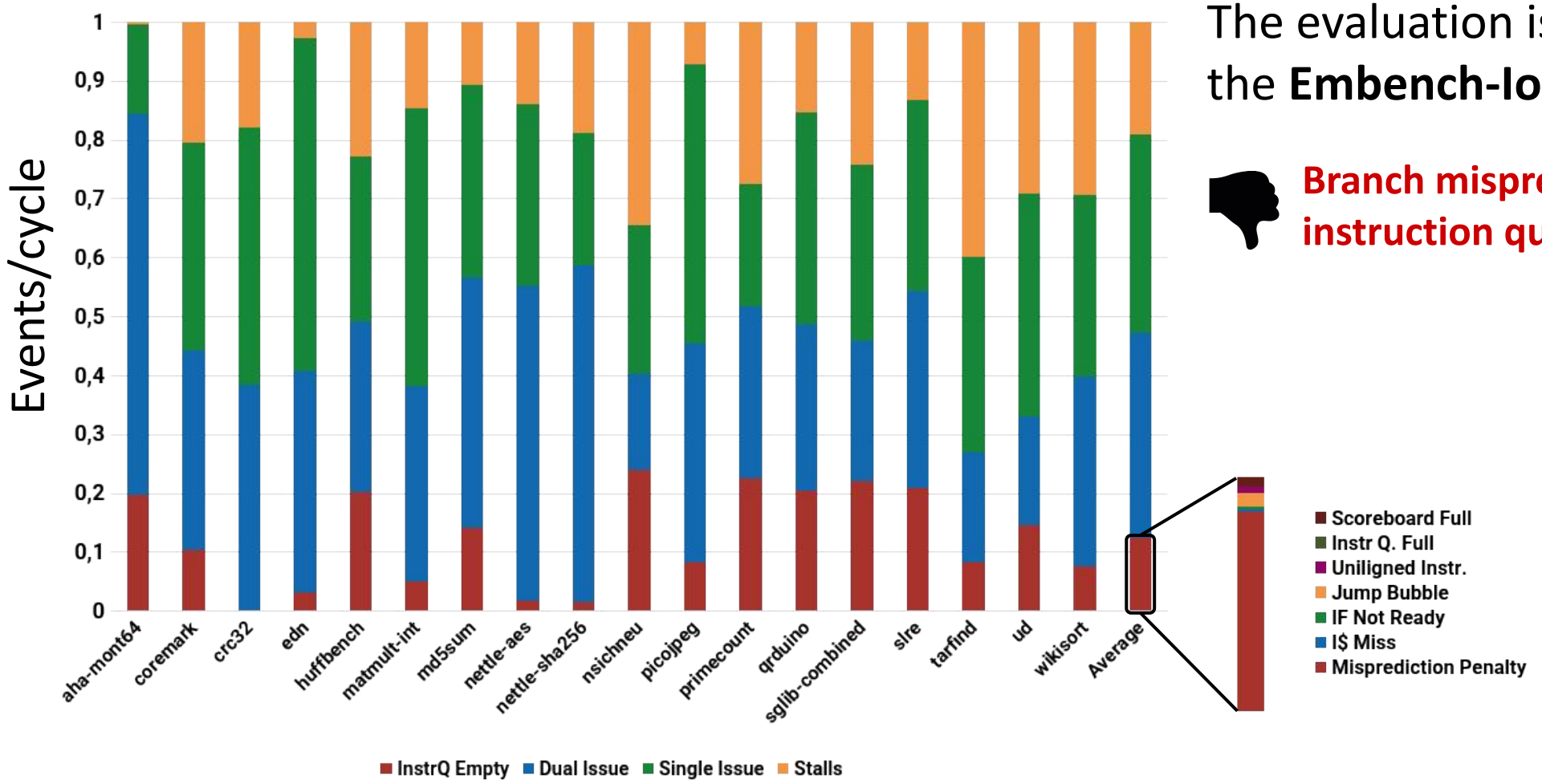
FPU support was **out of scope** for **CVA6S**

CVA6S: performance analysis



The evaluation is based on the **Embench-IoT suite**

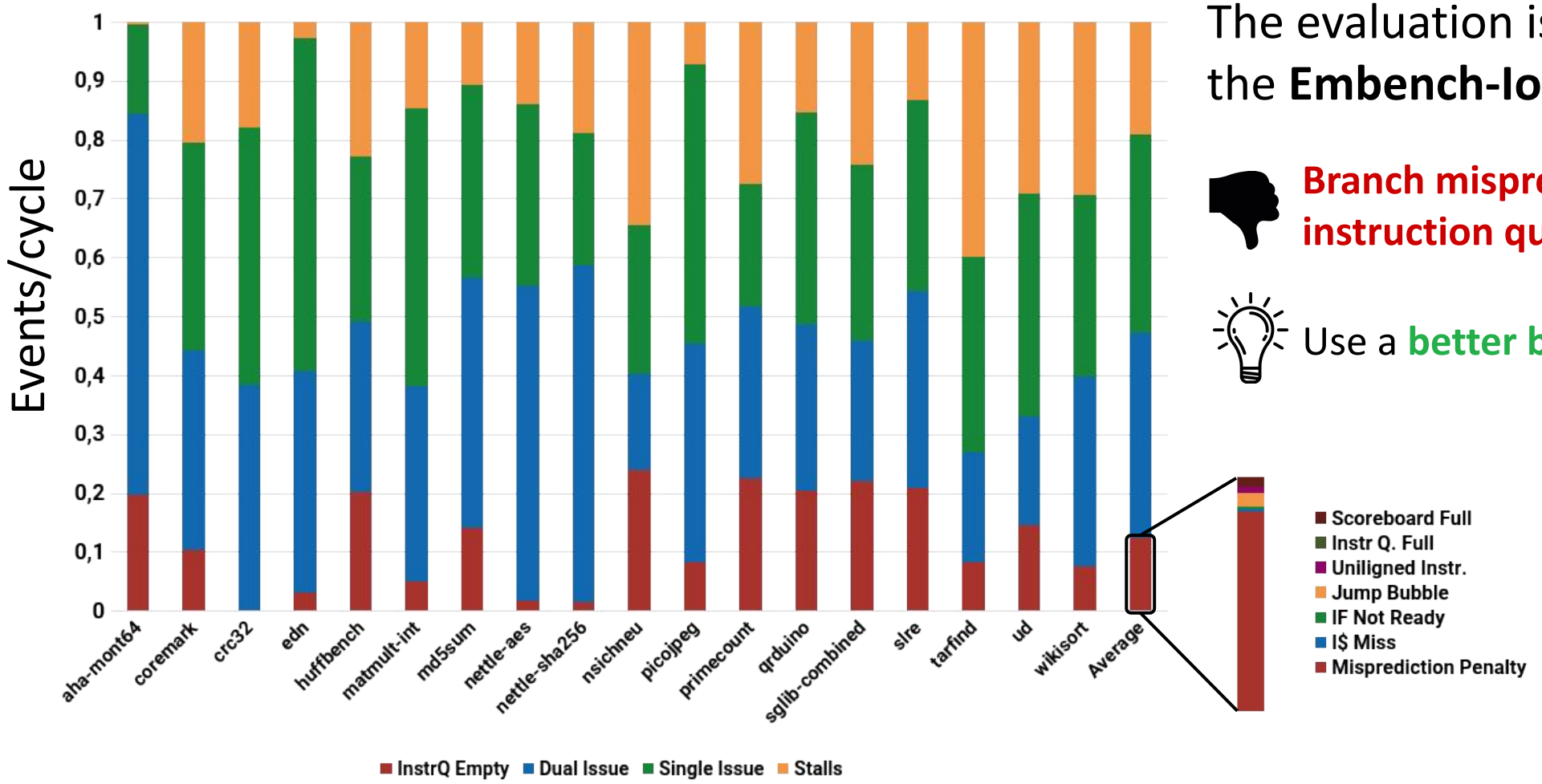
CVA6S: performance analysis



The evaluation is based on the **Embench-IoT suite**

Branch mispredictions cause the **instruction queue** to be **empty**

CVA6S: performance analysis

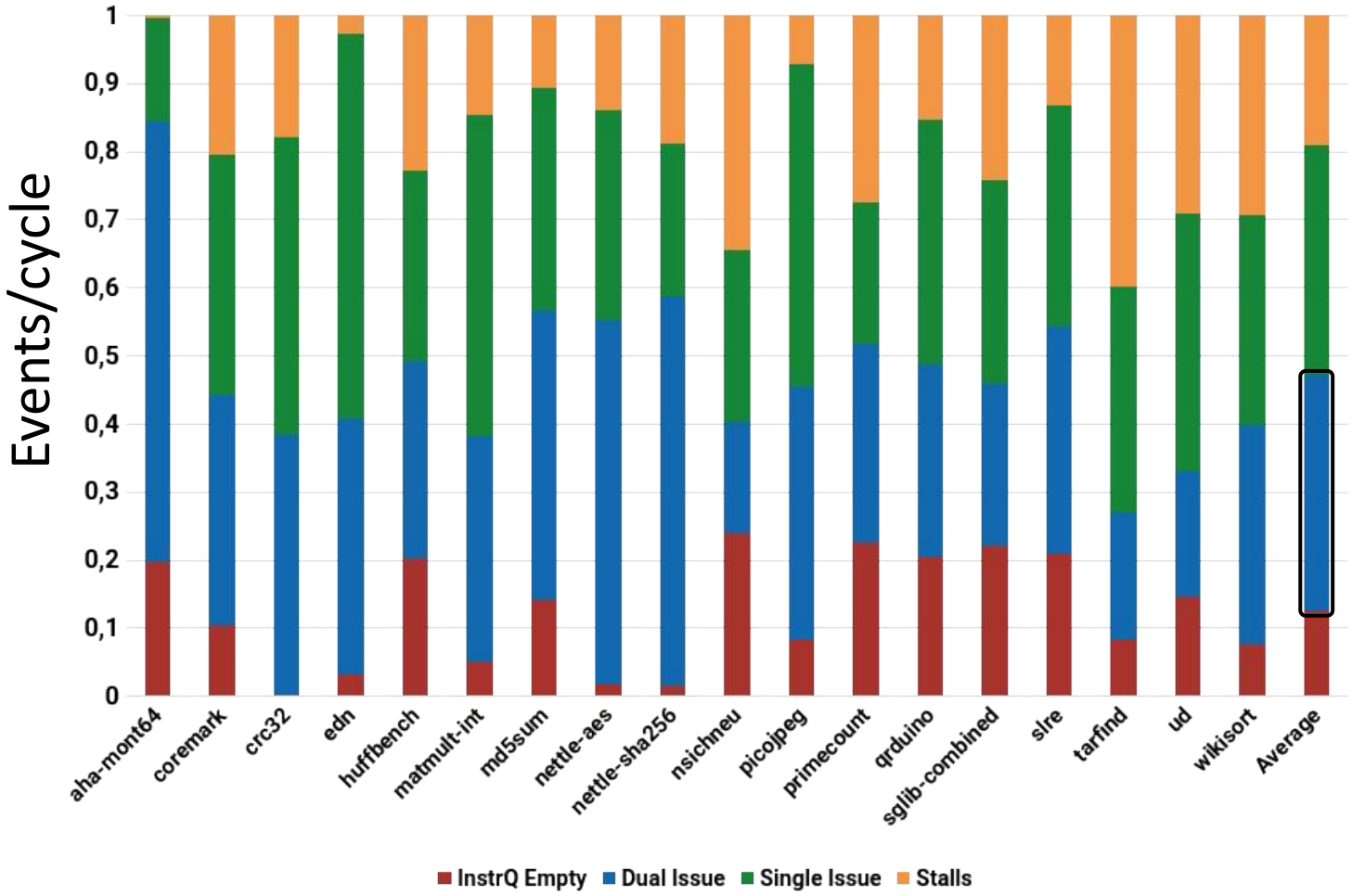


The evaluation is based on the **Embench-IoT suite**

Branch mispredictions cause the **instruction queue** to be **empty**

Use a **better branch predictor**

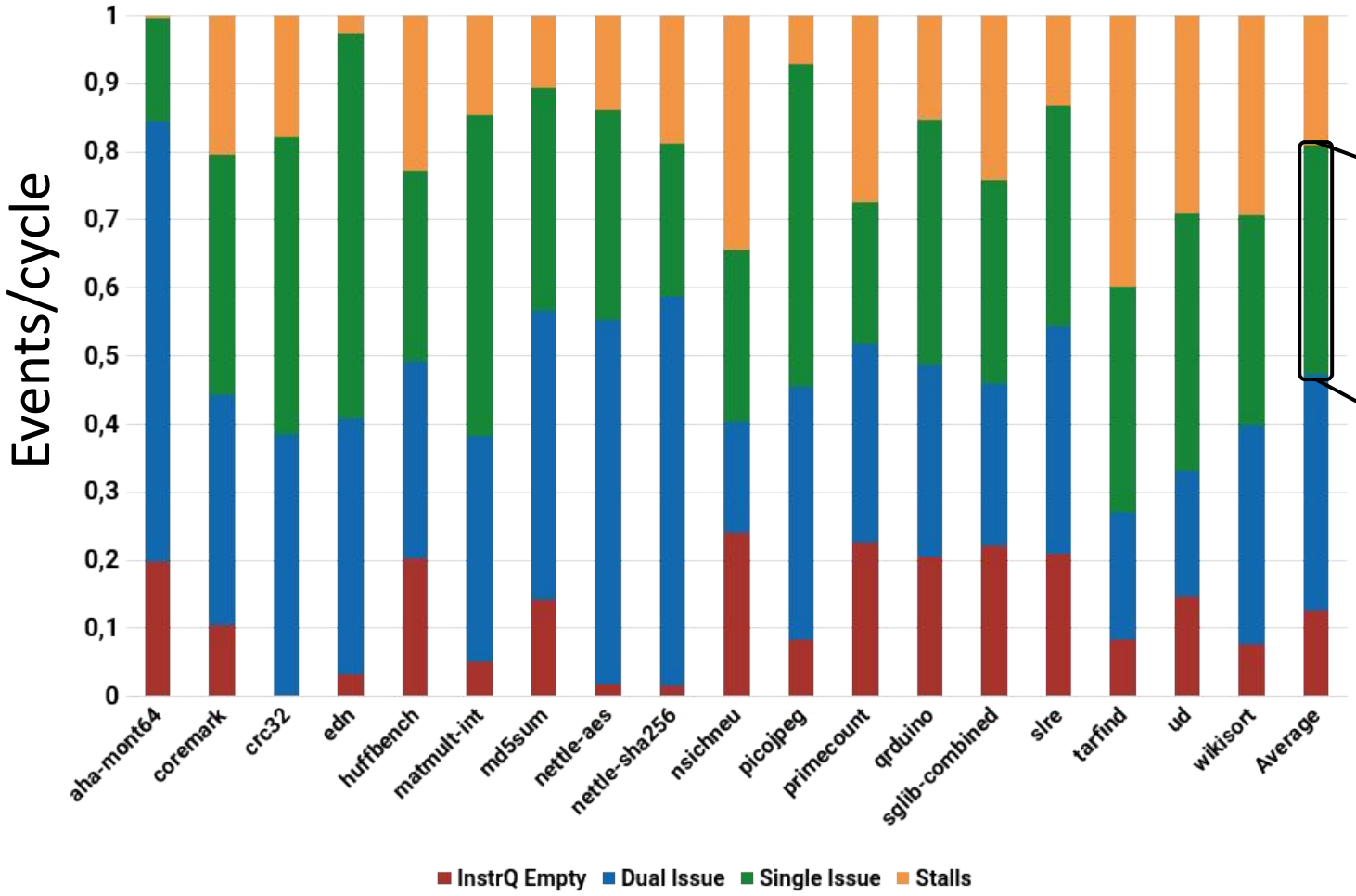
CVA6S: performance analysis



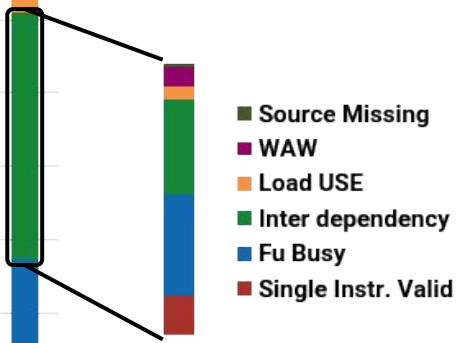
The evaluation is based on the **Embench-IoT suite**

Instructions are **dual issued** already for **30% of the cycles**

CVA6S: performance analysis

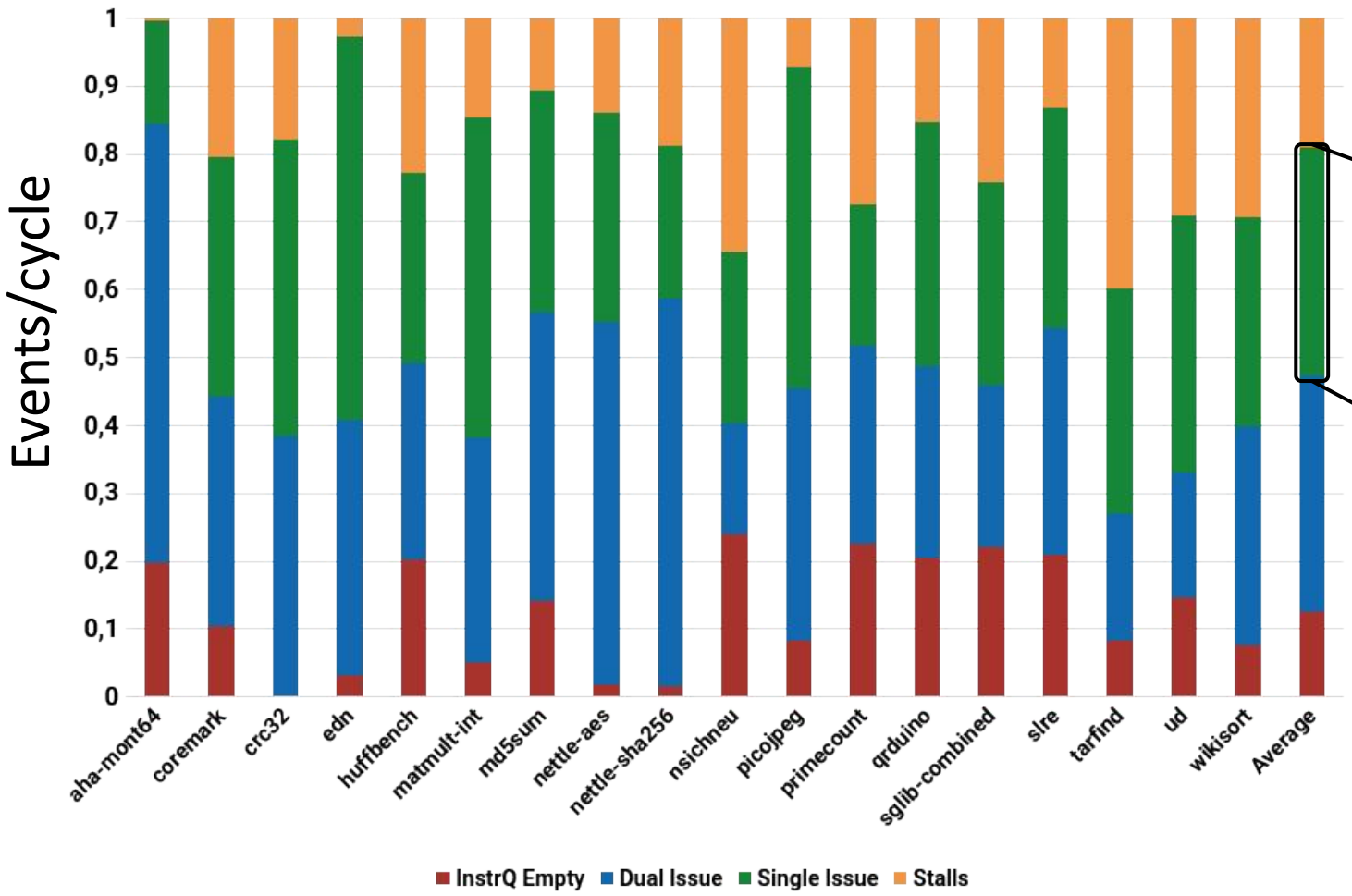


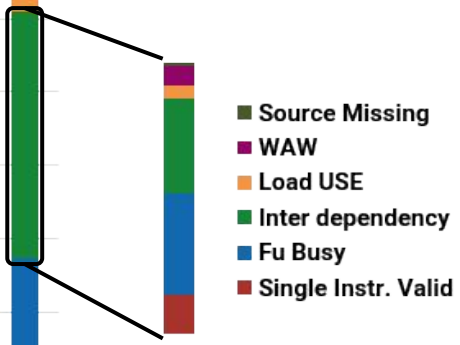
The evaluation is based on the **Embench-IoT suite**



Instruction dependencies cause **single-issuing**

CVA6S: performance analysis



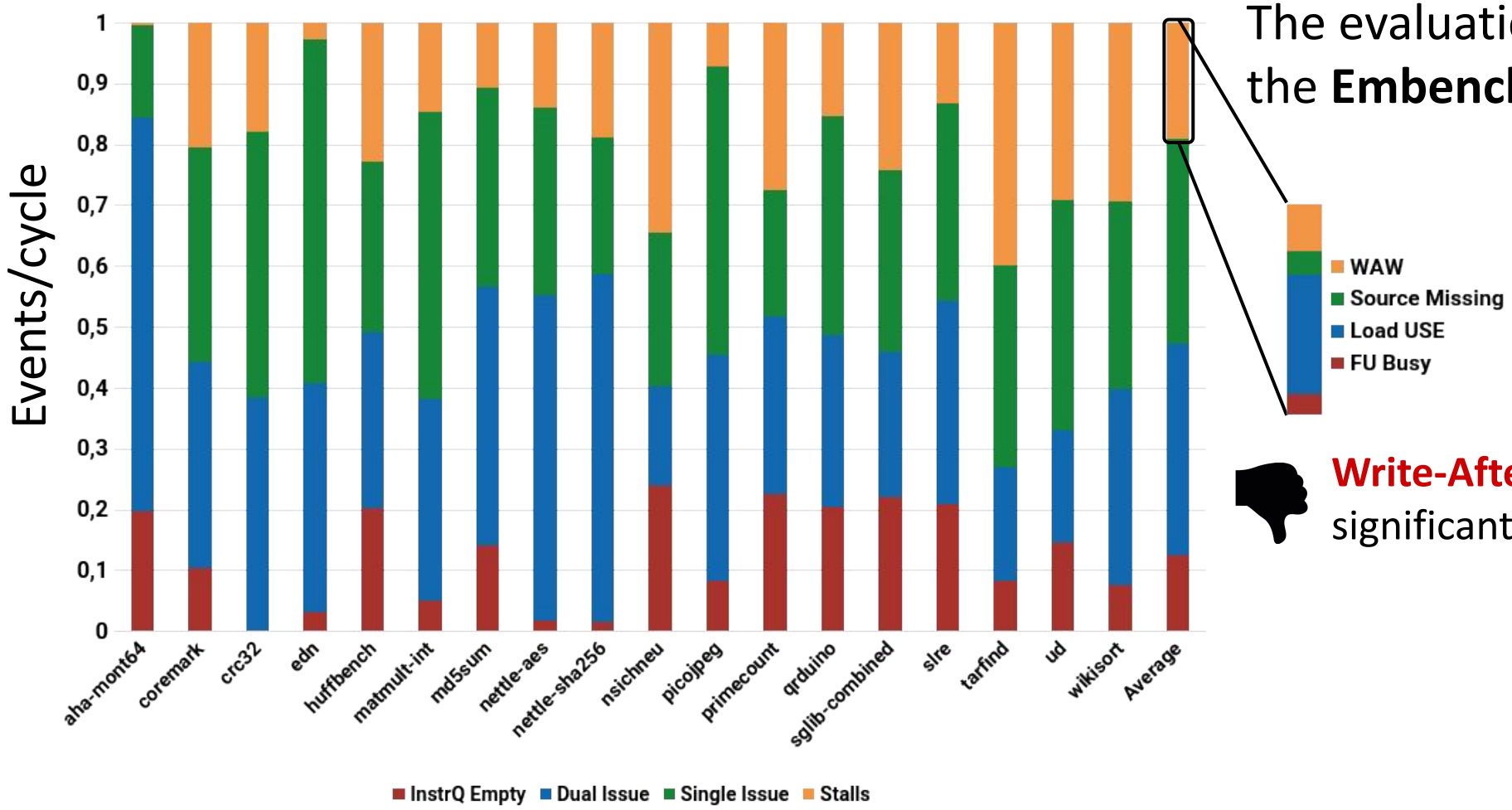


The evaluation is based on the **Embench-IoT suite**

Instruction dependencies cause **single-issuing**

Introduce **ALU-ALU forwarding** to **dual issue interdependent ALU operations**

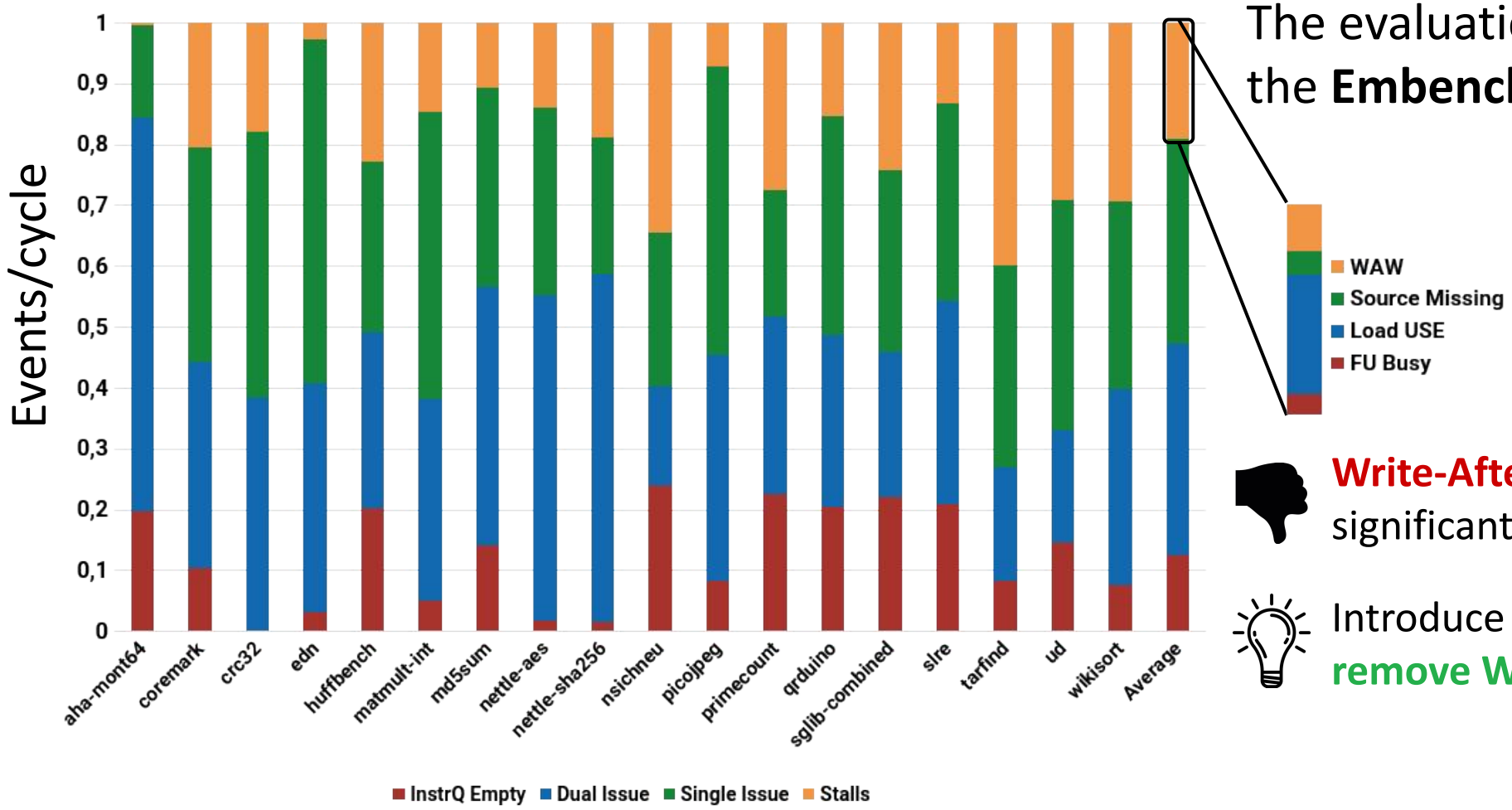
CVA6S: performance analysis



The evaluation is based on the **Embench-IoT suite**

Write-After-Write hazards are a significant cause of **pipeline stalls**

CVA6S: performance analysis



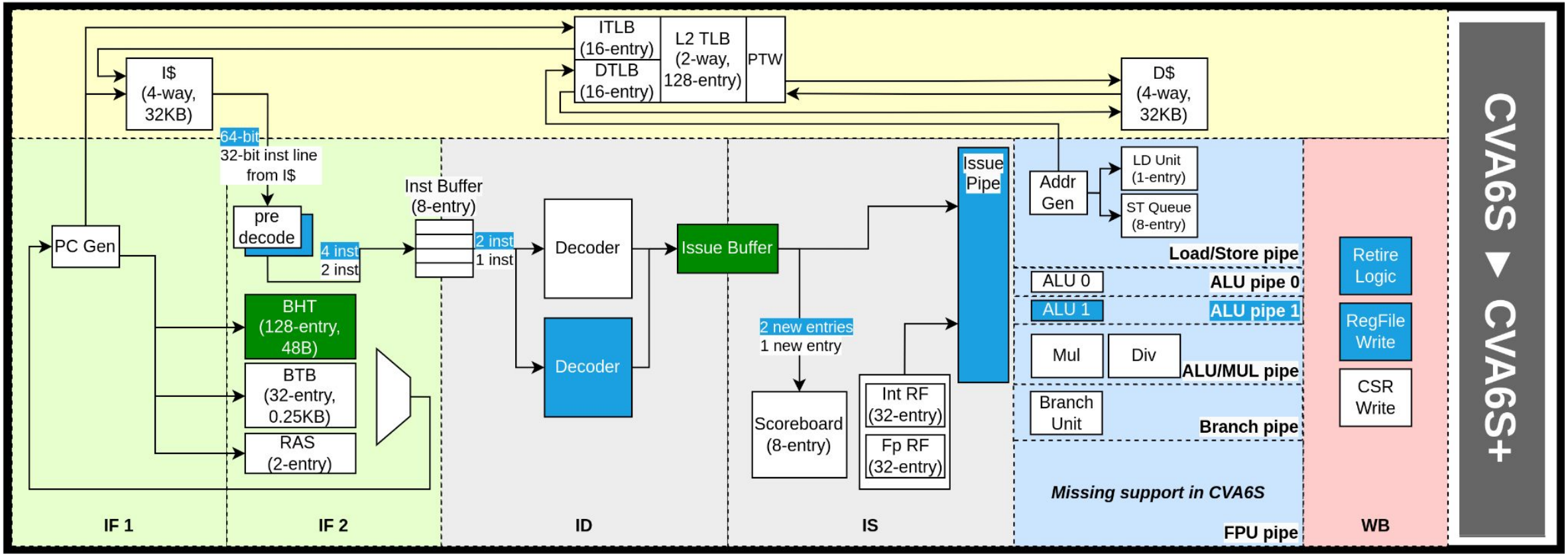
The evaluation is based on the **Embench-IoT suite**

Write-After-Write hazards are a significant cause of **pipeline stalls**

Introduce **register renaming** to **remove WAWs**

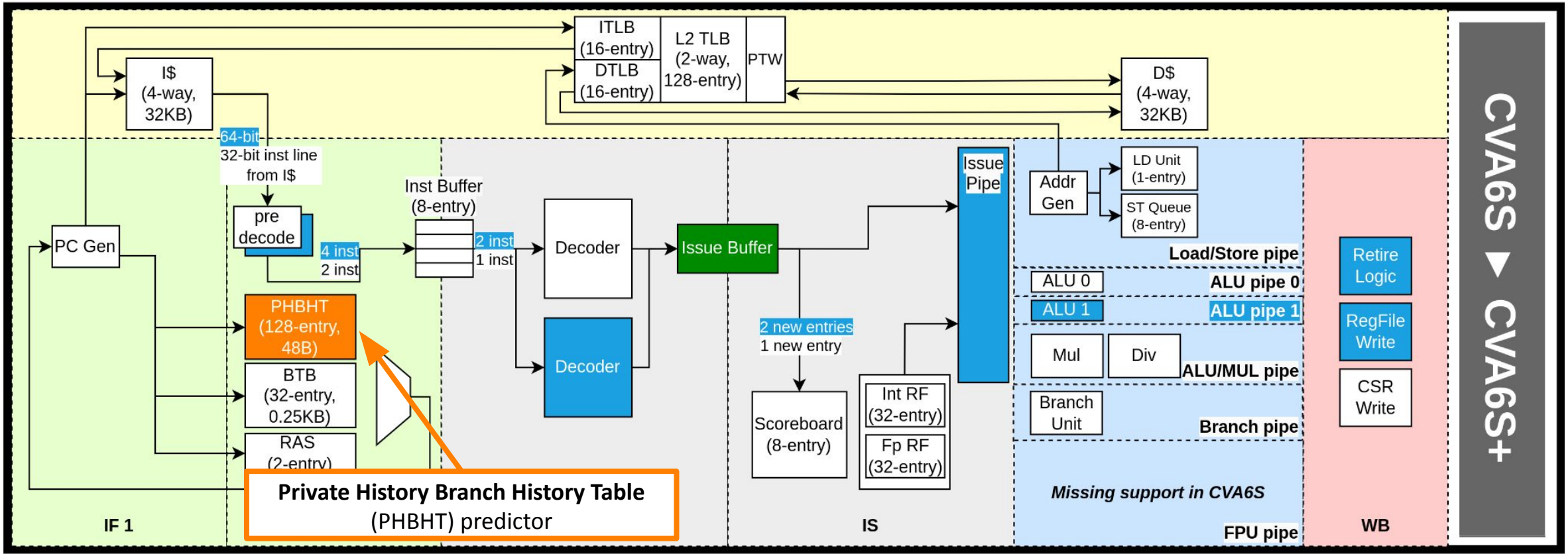
CVA6S+: what's new?

Wider **Bigger**



CVA6S+: Private History Predictor

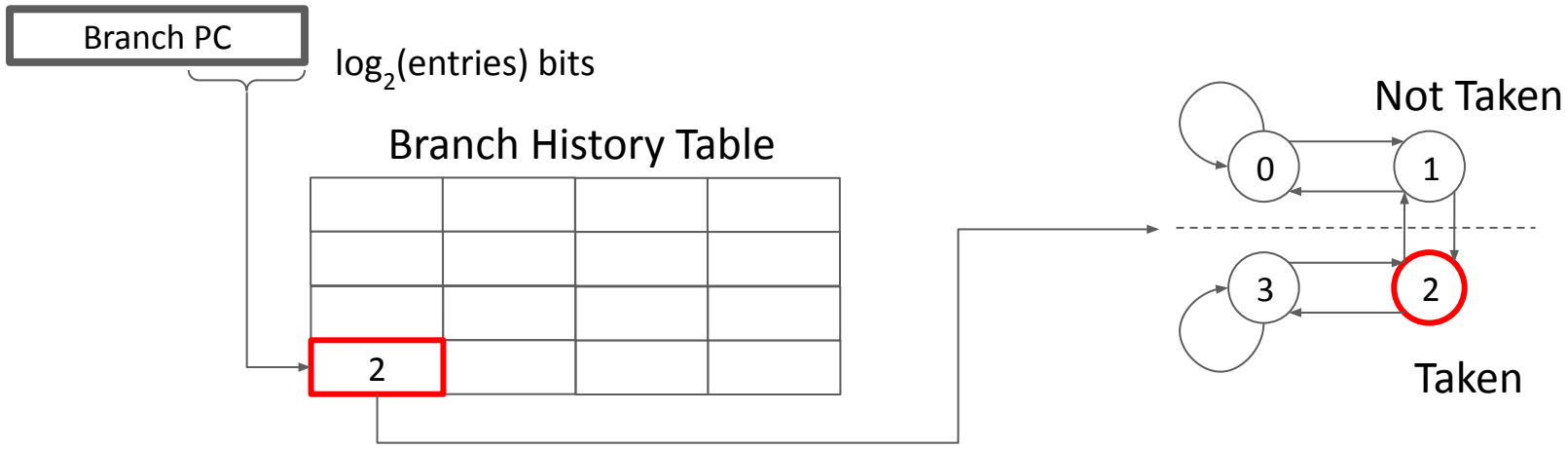
Wider **Bigger** **New**



CVA6S+: Private History Predictor

Legacy BHT predictor

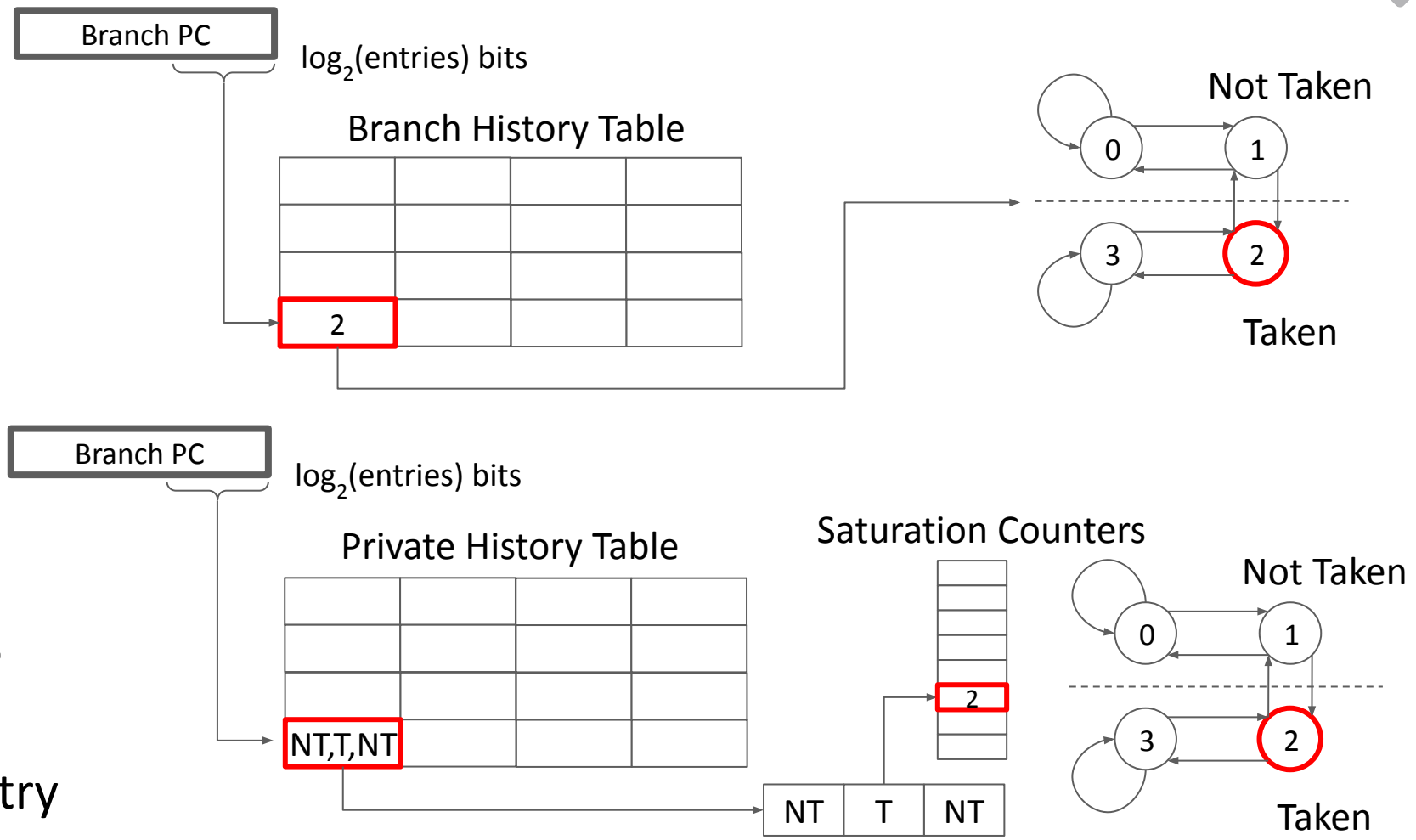
2-bit per entry



CVA6S+: Private History Predictor

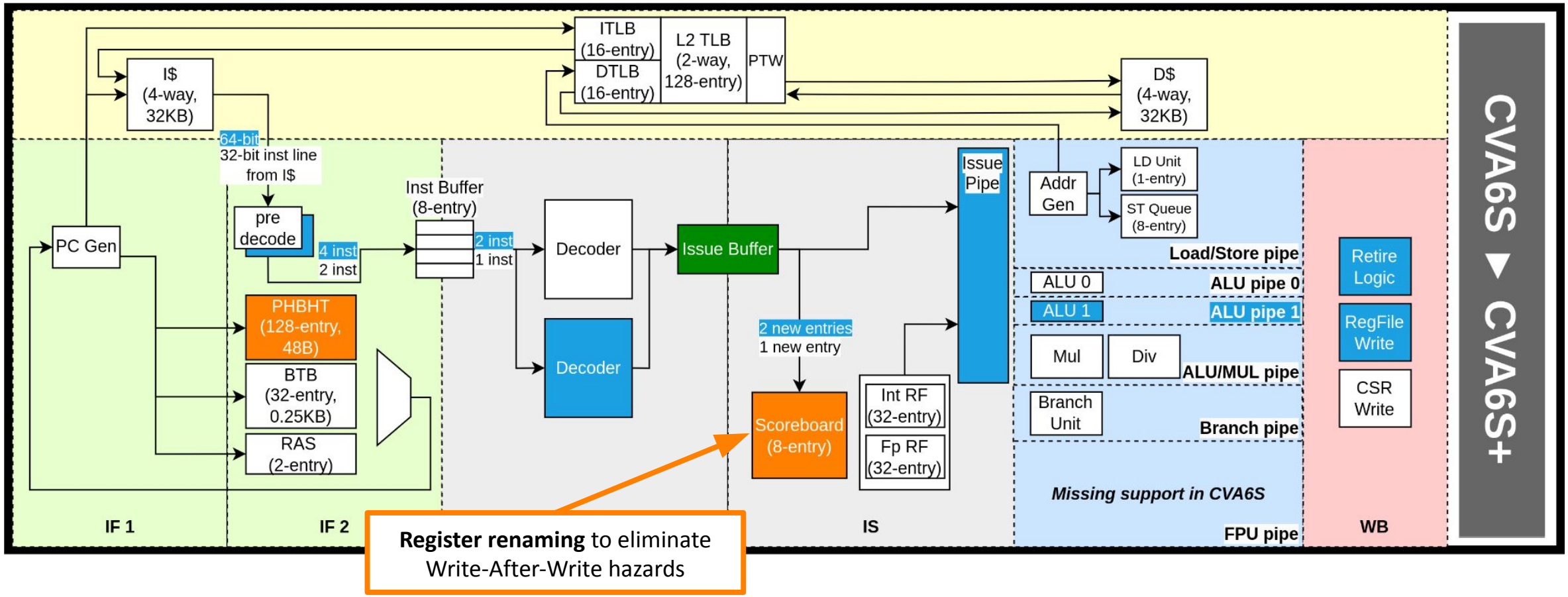
Legacy BHT predictor
2-bit per entry

New PHBHT predictor
with n bits history
$n + (2^n * 2)$ bits per entry



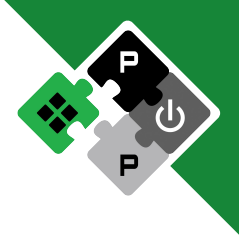
CVA6S+: Renaming Scheme

Wider **Bigger** **New**



Register renaming to eliminate Write-After-Write hazards

CVA6S+: Renaming Scheme

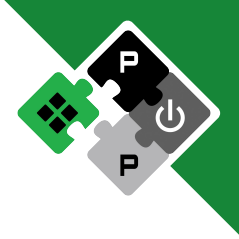


- ➔ The **scoreboard** is a **circular buffer**
- ➔ **RAW hazards** need to know the **newest instruction** to correctly **forward data**

Scoreboard (SB)

ID	Valid	rd	
7	1	10	
6	1	11	
5	1	12	
4	0		
3	0		
2	0		
1	1	5	
0	1	12	

CVA6S+: Renaming Scheme



- ➔ The **scoreboard** is a **circular buffer**
- ➔ **RAW hazards** need to know the **newest instruction** to correctly **forward data**

Scoreboard (SB)

ID	Valid	rd	
7	1	10	
6	1	11	
5	1	12	
4	0		
3	0		
2	0		
1	1	5	
0	1	12	

Instr. 0 and 5 both write register x12

CVA6S+: Renaming Scheme

- ➔ The **scoreboard** is a **circular buffer**
- ➔ **RAW hazards** need to know the **newest instruction** to correctly **forward data**

Reorder the scoreboard based on commit pointer

Scoreboard (SB)

ID	Valid	rd	
7	1	10	
6	1	11	
5	1	12	
4	0		
3	0		
2	0		
1	1	5	
0	1	12	

0	
0	
0	
1	SB-ID: 1
1	SB-ID: 0
1	SB-ID: 7
1	SB-ID: 6
1	SB-ID: 5

Instr. 0 and 5 both write register x12

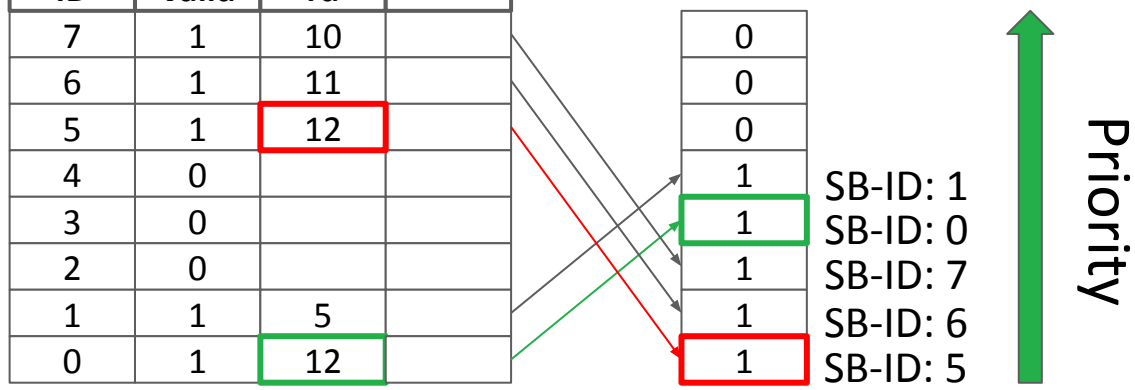
CVA6S+: Renaming Scheme

- ➔ The **scoreboard** is a **circular buffer**
- ➔ **RAW hazards** need to know the **newest instruction** to correctly **forward data**

Reorder the scoreboard based on commit pointer

Scoreboard (SB)

ID	Valid	rd	
7	1	10	
6	1	11	
5	1	12	
4	0		
3	0		
2	0		
1	1	5	
0	1	12	



Instr. 0 and 5 both write register x12

Instr. 0 has higher priority than instr. 5

CVA6S+: Renaming Scheme

- ➔ The **scoreboard** is a **circular buffer**
- ➔ **RAW hazards** need to know the **newest instruction** to correctly **forward data**

Scoreboard (SB)

ID	Valid	rd	
7	1	10	
6	1	11	
5	1	12	
4	0		
3	0		
2	0		
1	1	5	
0	1	12	

Instr. 0 and 5 both write register x12

Reorder the scoreboard based on commit pointer

0	SB-ID: 1
0	SB-ID: 0
0	SB-ID: 7
1	SB-ID: 6
1	SB-ID: 5
1	
1	
1	

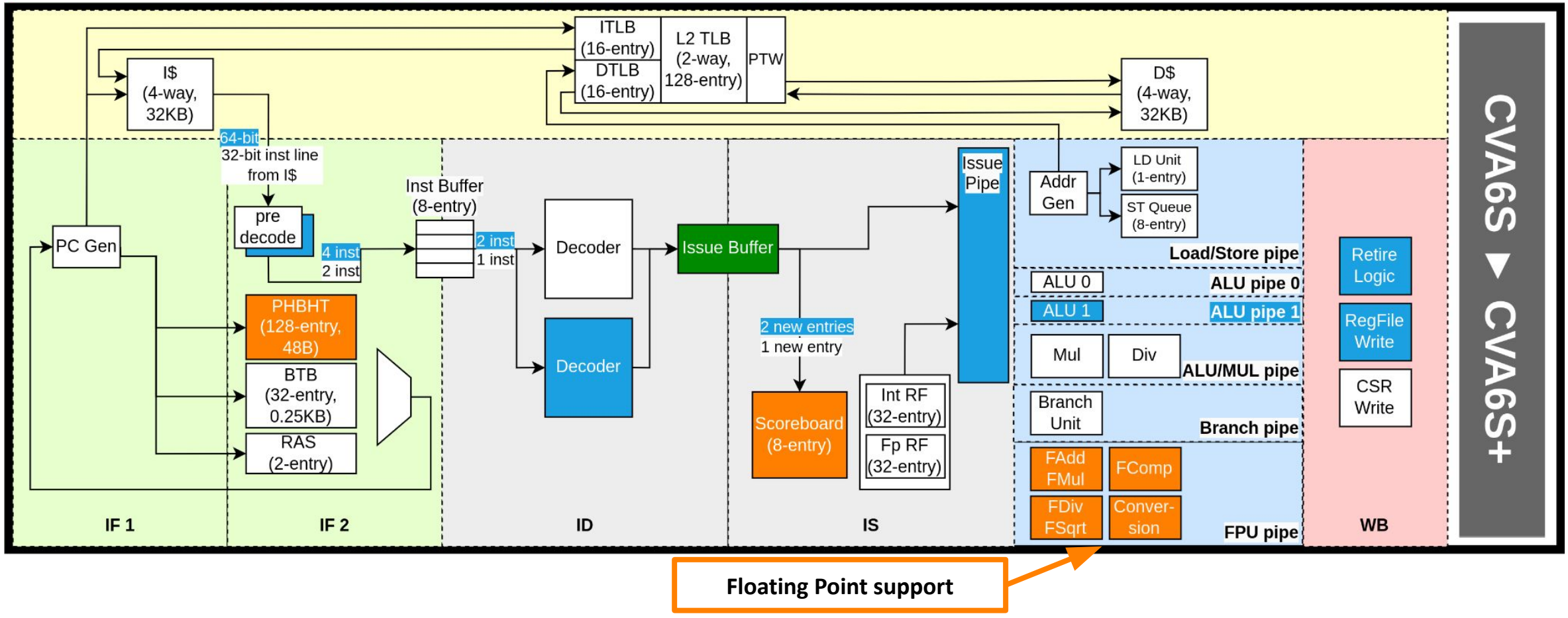
Instr. 0 has higher priority than instr. 5

Forwarding logic based on SB-ID

-	x31
0	x12
6	x11
7	x10
-	x9
-	x8
-	x7
-	x6
1	x5
-	x4
-	x3
-	x2
-	x1
-	x0

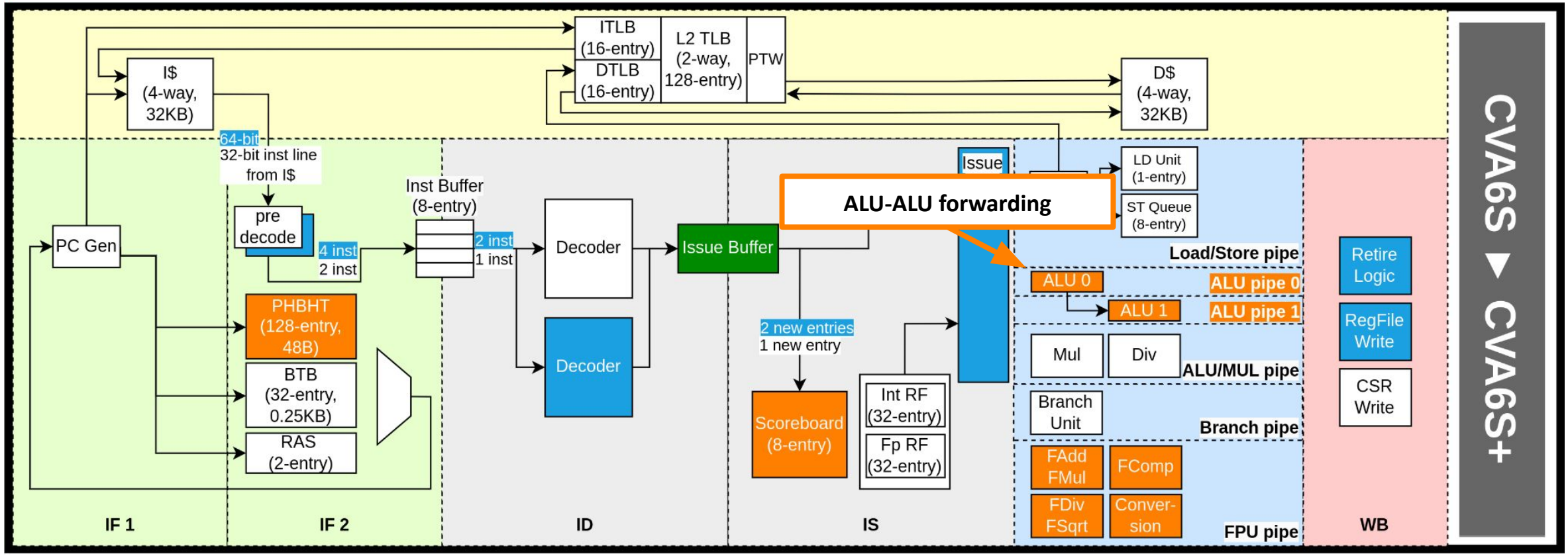
CVA6S+: FPU support

Wider **Bigger** **New**



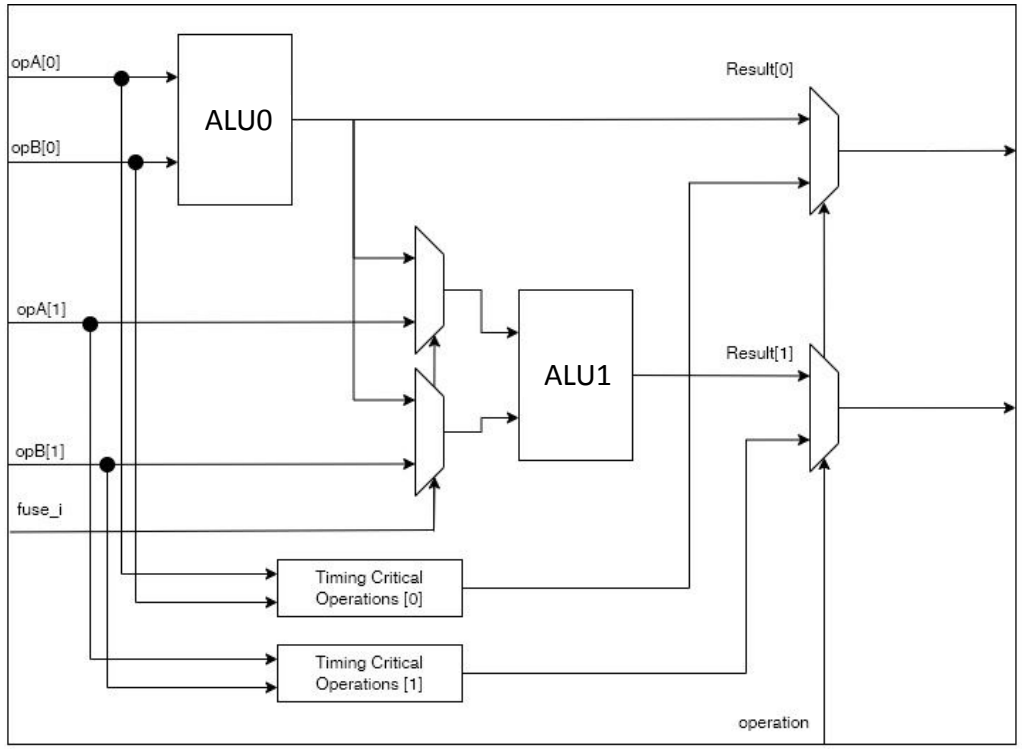
CVA6S+: ALU-ALU forwarding

Wider **Bigger** **New**



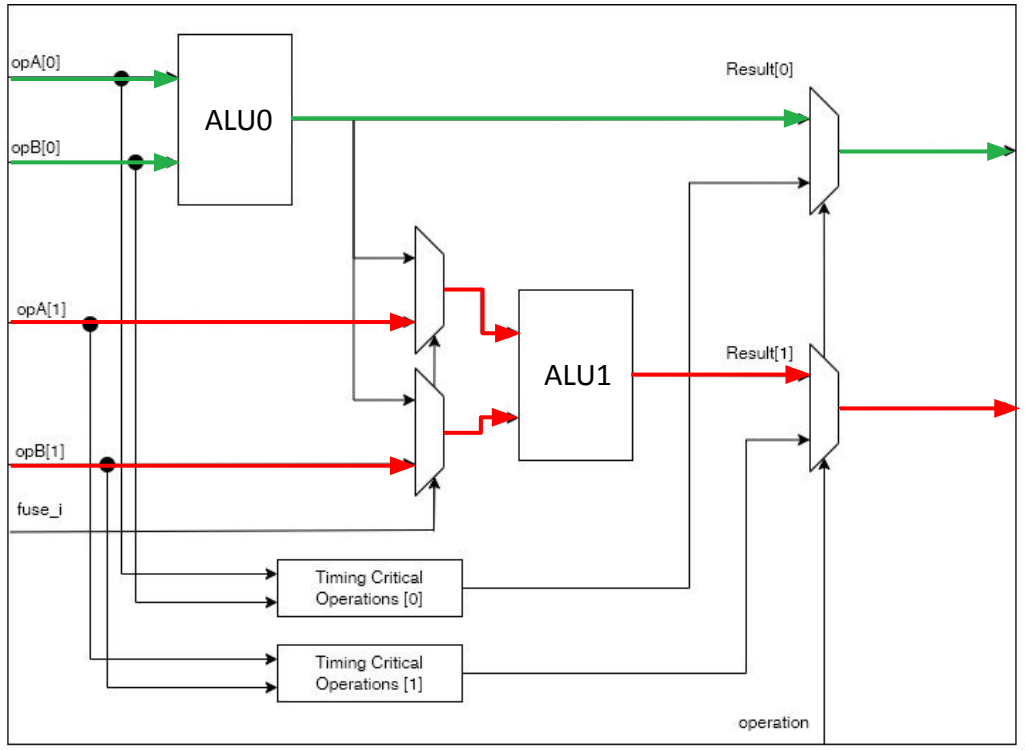
CVA6S+: ALU-ALU forwarding

➔ The ALUs operate separately when dual-issuing independent instructions



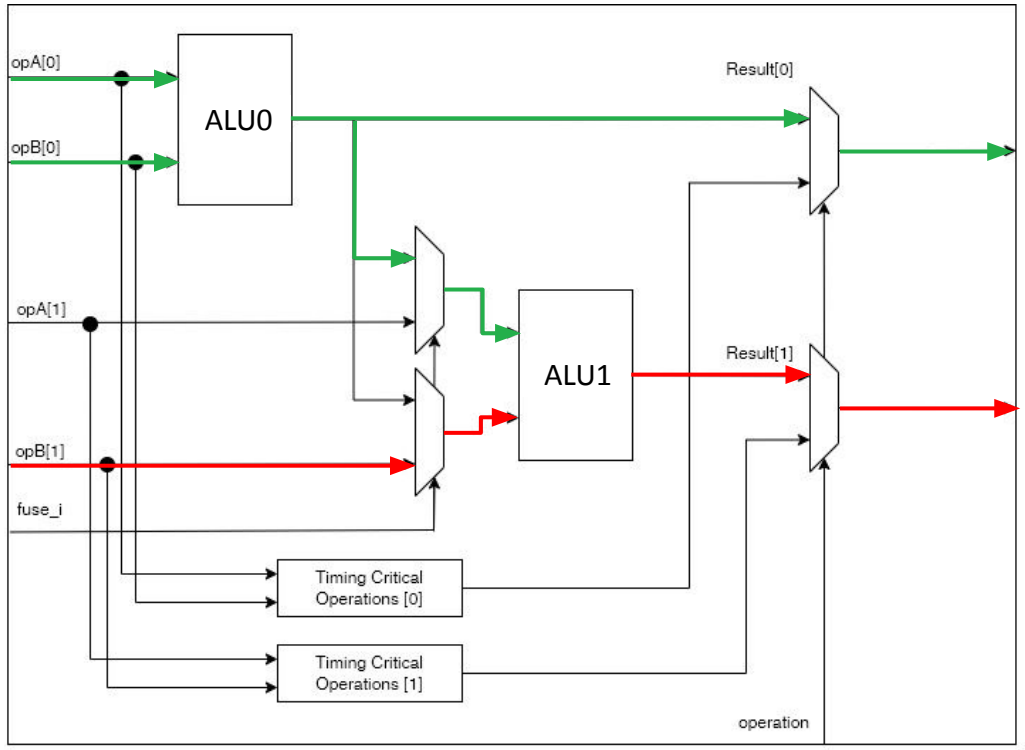
CVA6S+: ALU-ALU forwarding

➔ The ALUs operate **separately** when dual-issuing independent instructions



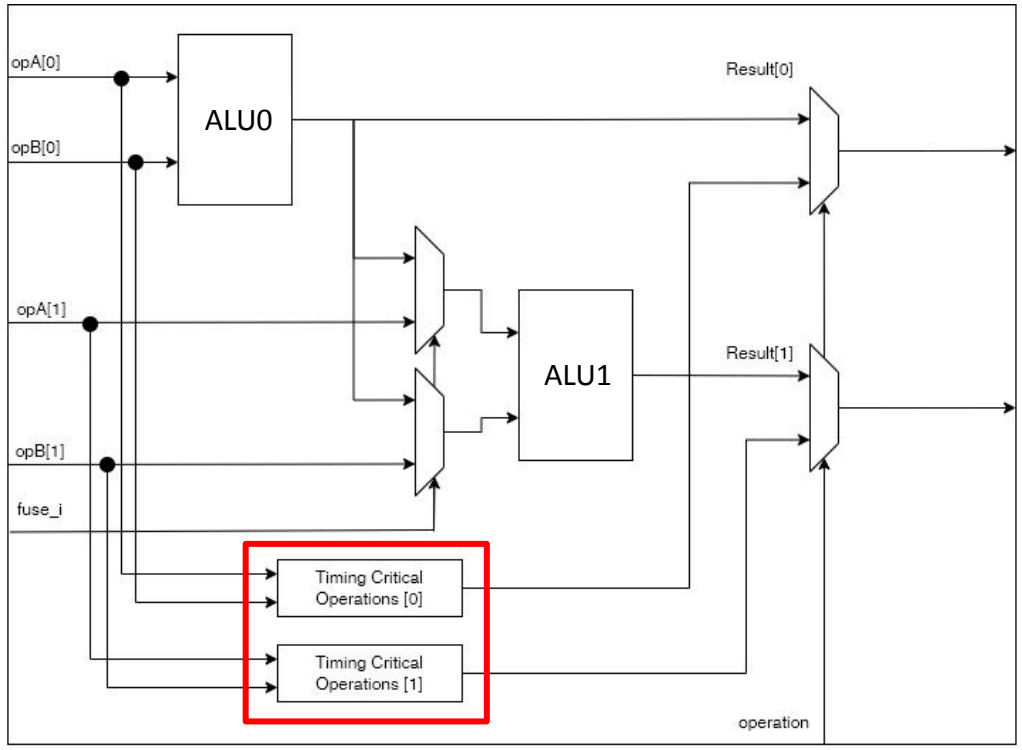
CVA6S+: ALU-ALU forwarding

- ➔ The ALUs operate separately when dual-issuing independent instructions
- ➔ The ALUs are **chained** when dependent instructions can be fused

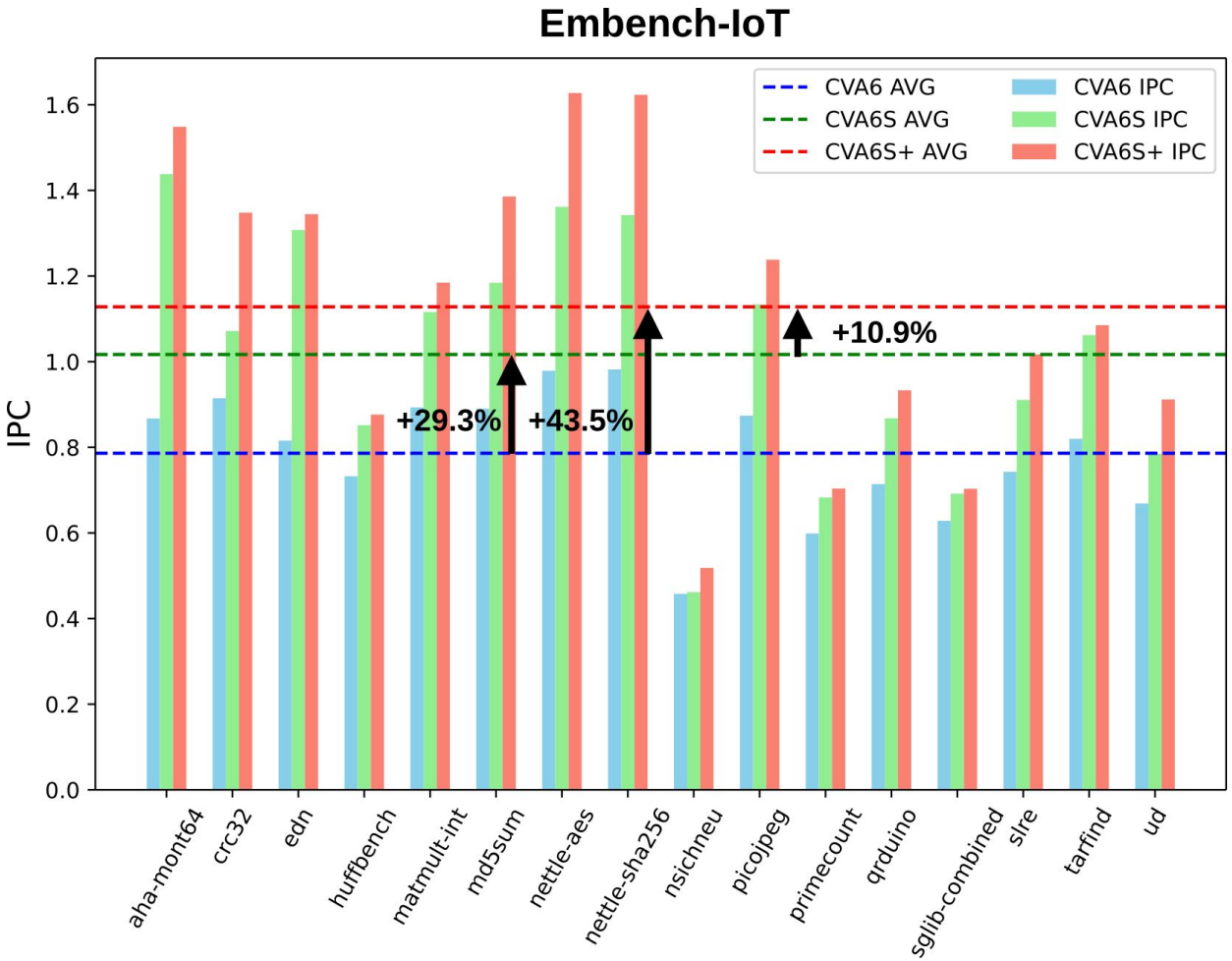


CVA6S+: ALU-ALU forwarding

- ➔ The ALUs operate separately when dual-issuing independent instructions
- ➔ The ALUs are chained when dependent instructions can be fused
- ➔ Selected **few operations** are **never chained** to **preserve the critical path**

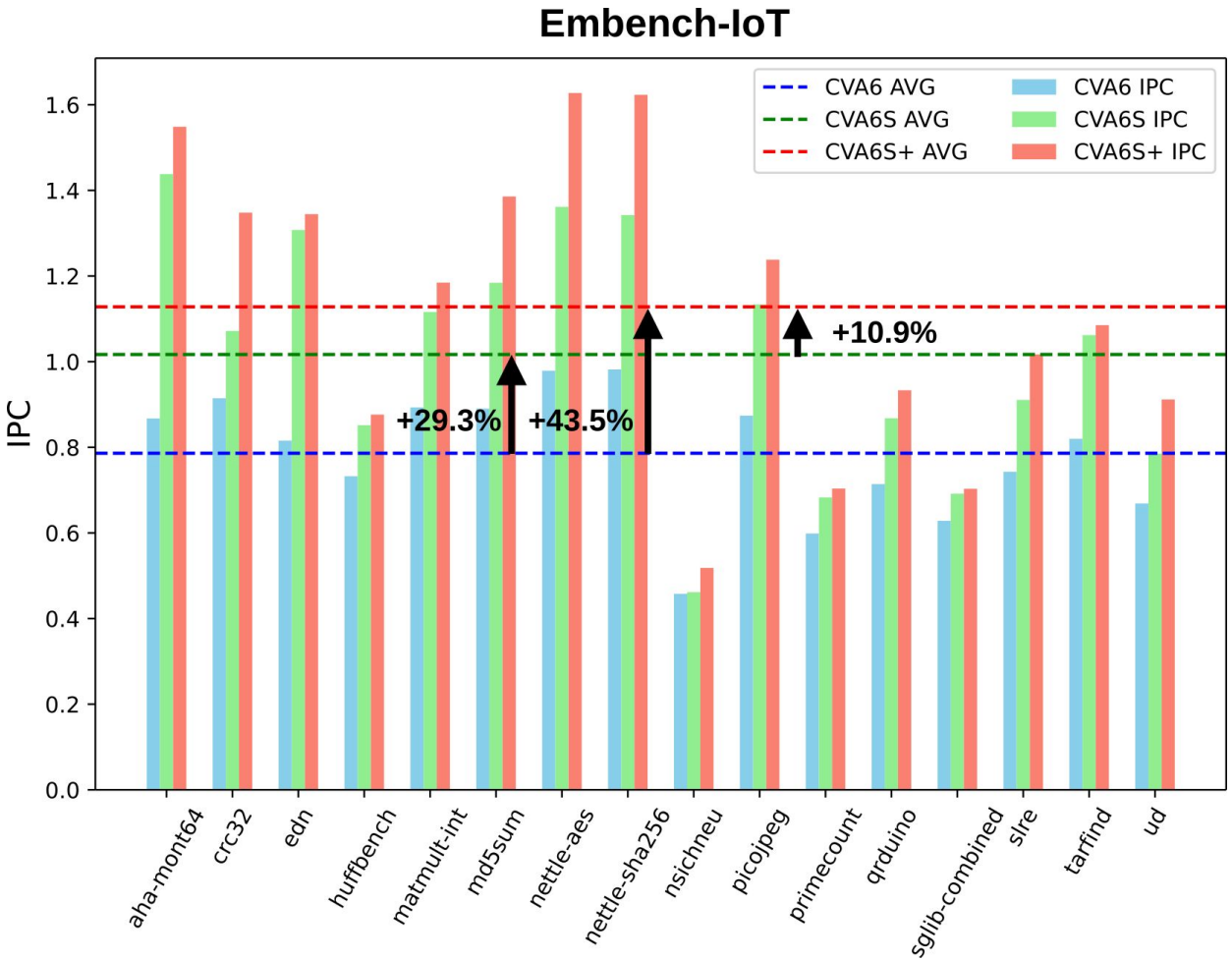


Pipeline performance: Embench-IoT



The **Embench-IoT suite** focuses on the **pipeline:**

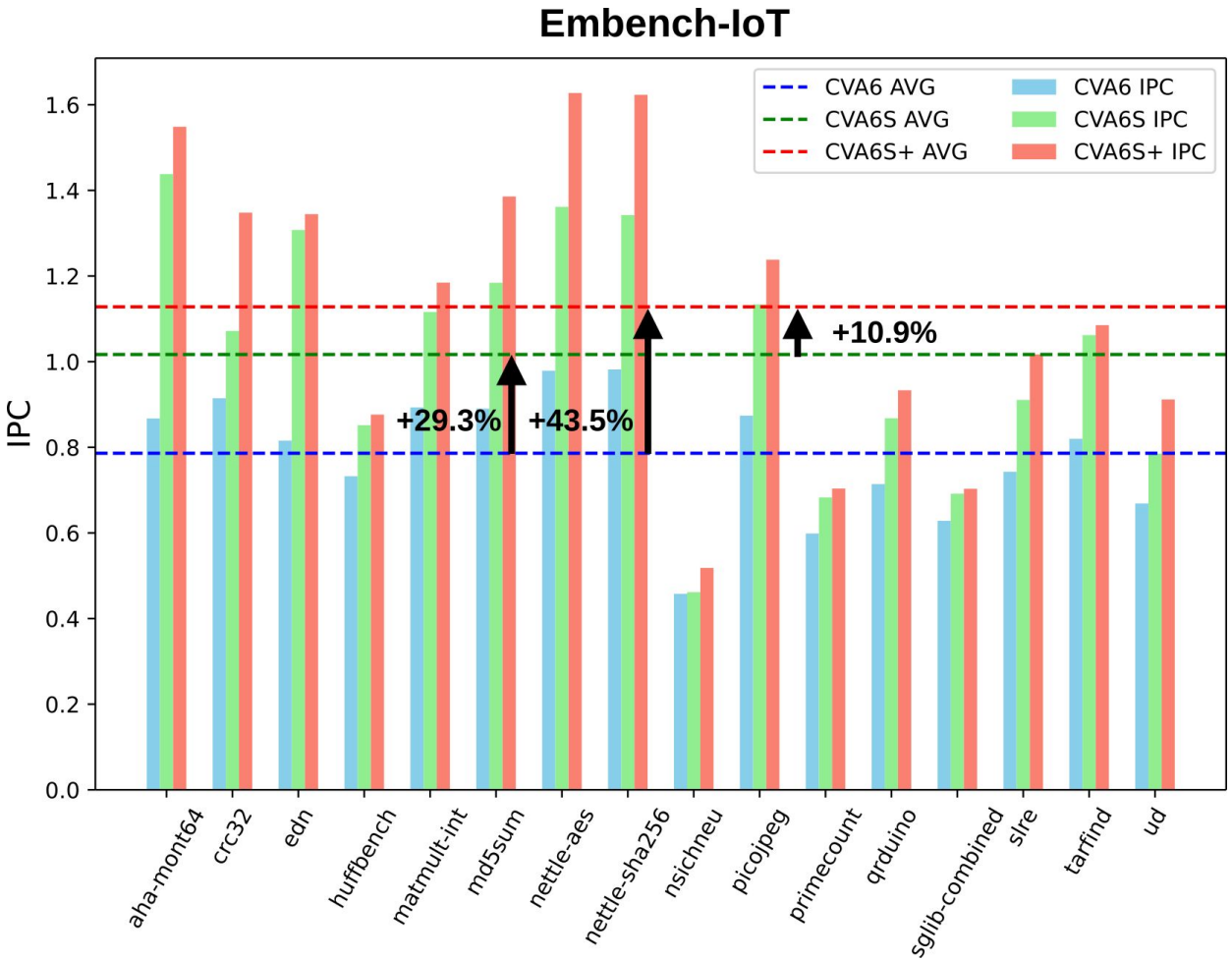
Pipeline performance: Embench-IoT



The **Embench-IoT suite** focuses on the **pipeline:**

➔ All the cores use the **same cache configuration**

Pipeline performance: Embench-IoT

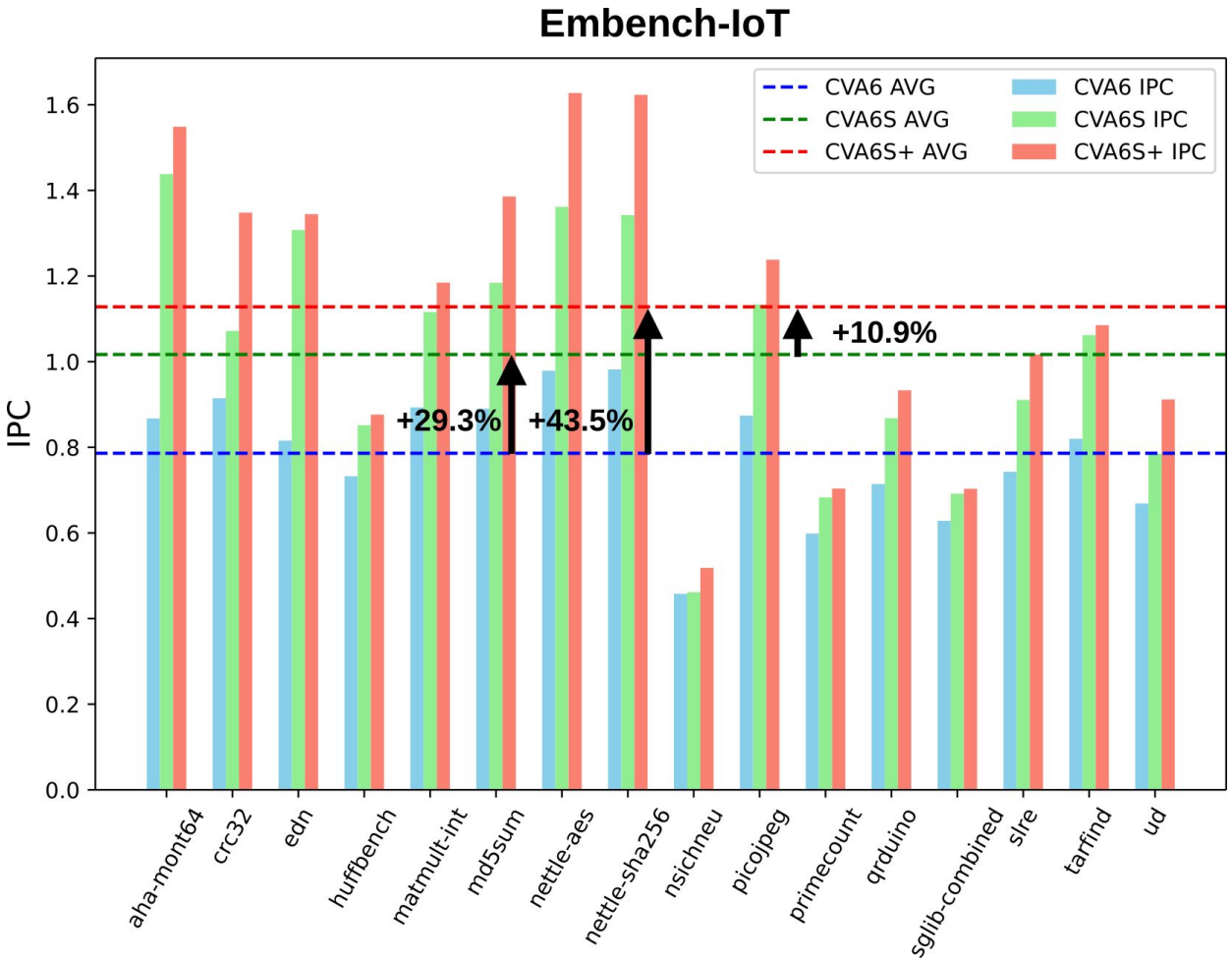


The **Embench-IoT suite** focuses on the **pipeline:**

➔ All the cores use the **same cache configuration**

➔ The **working set** is **fully cached**

Pipeline performance: Embench-IoT



The **Embench-IoT suite** focuses on the **pipeline:**

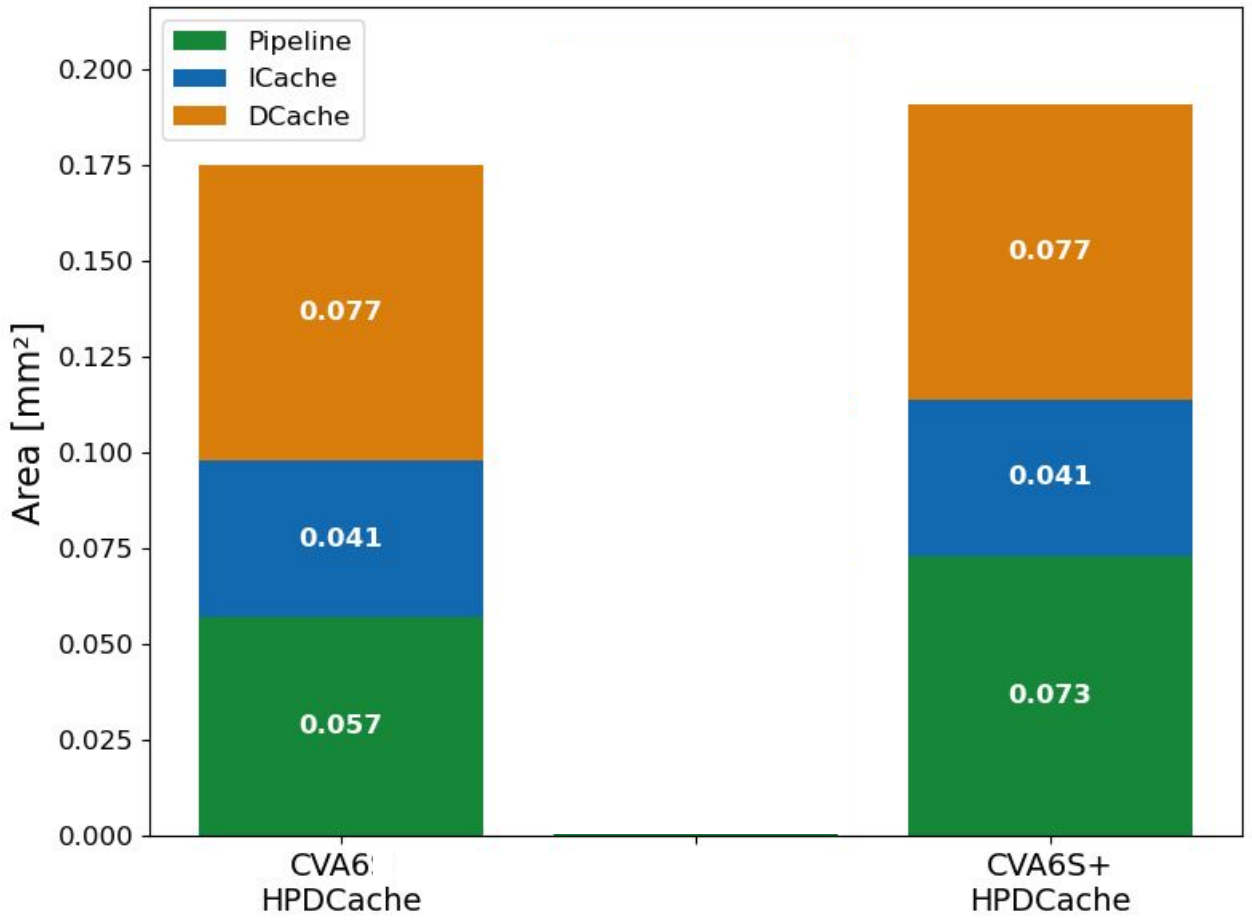
➔ All the cores use the **same cache configuration**

➔ The **working set** is **fully cached**

+43.5% IPC versus **baseline CVA6**

+10.9% IPC versus **CVA6S**

Area and Timing: performance at what cost?

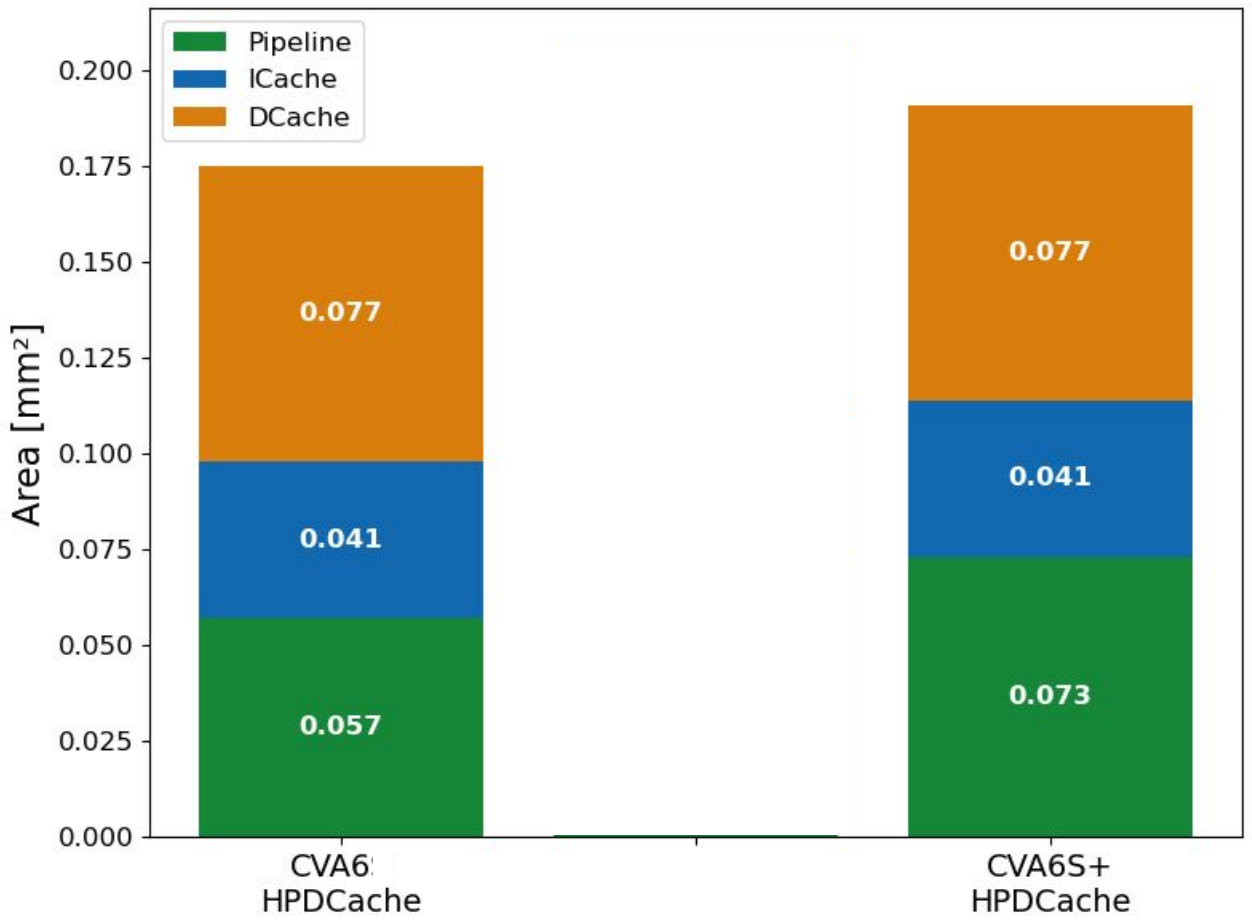


Evaluation setup:

➔ **GF22 nm** CMOS technology

➔ Worst timing corner

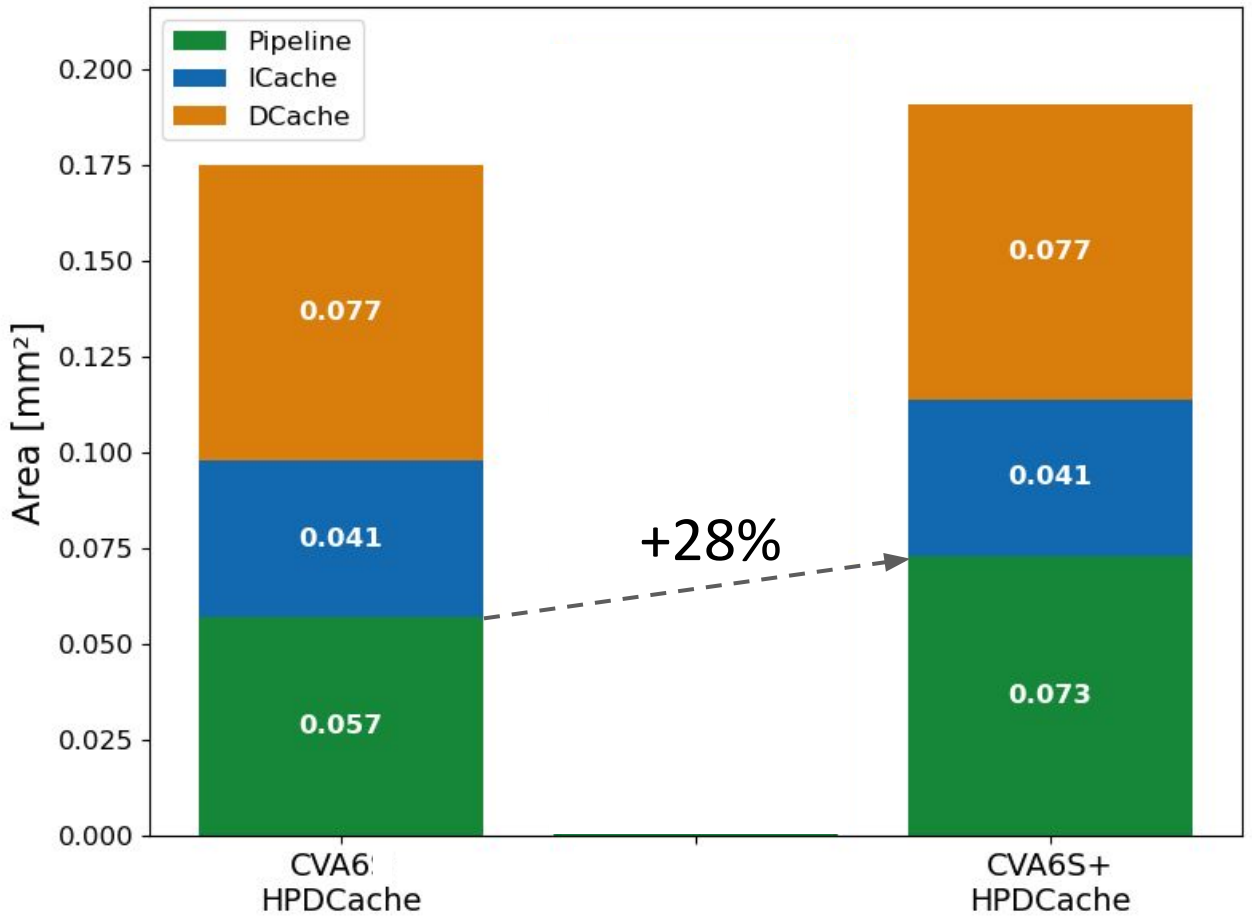
Area and Timing: performance at what cost?



Evaluation setup:

- ➔ **GF22 nm** CMOS technology
- ➔ Worst timing corner
- ➔ **CVA6S+ versus CVA6**
- ➔ **Same caches** configuration

Area and Timing: performance at what cost?

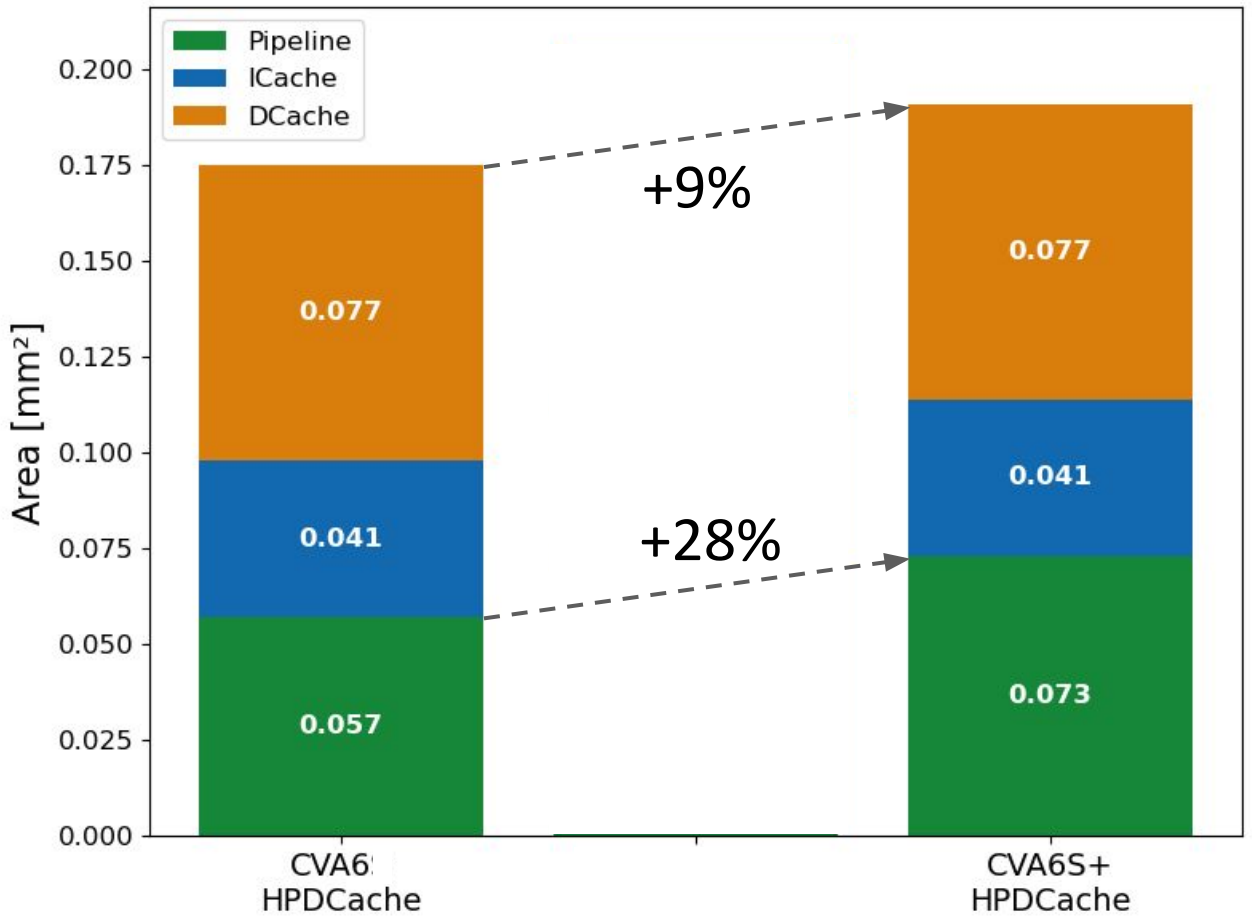


Evaluation setup:

- ➔ **GF22 nm** CMOS technology
- ➔ Worst timing corner
- ➔ **CVA6S+ versus CVA6**
- ➔ **Same caches** configuration

Pipeline area delta: +28%

Area and Timing: performance at what cost?



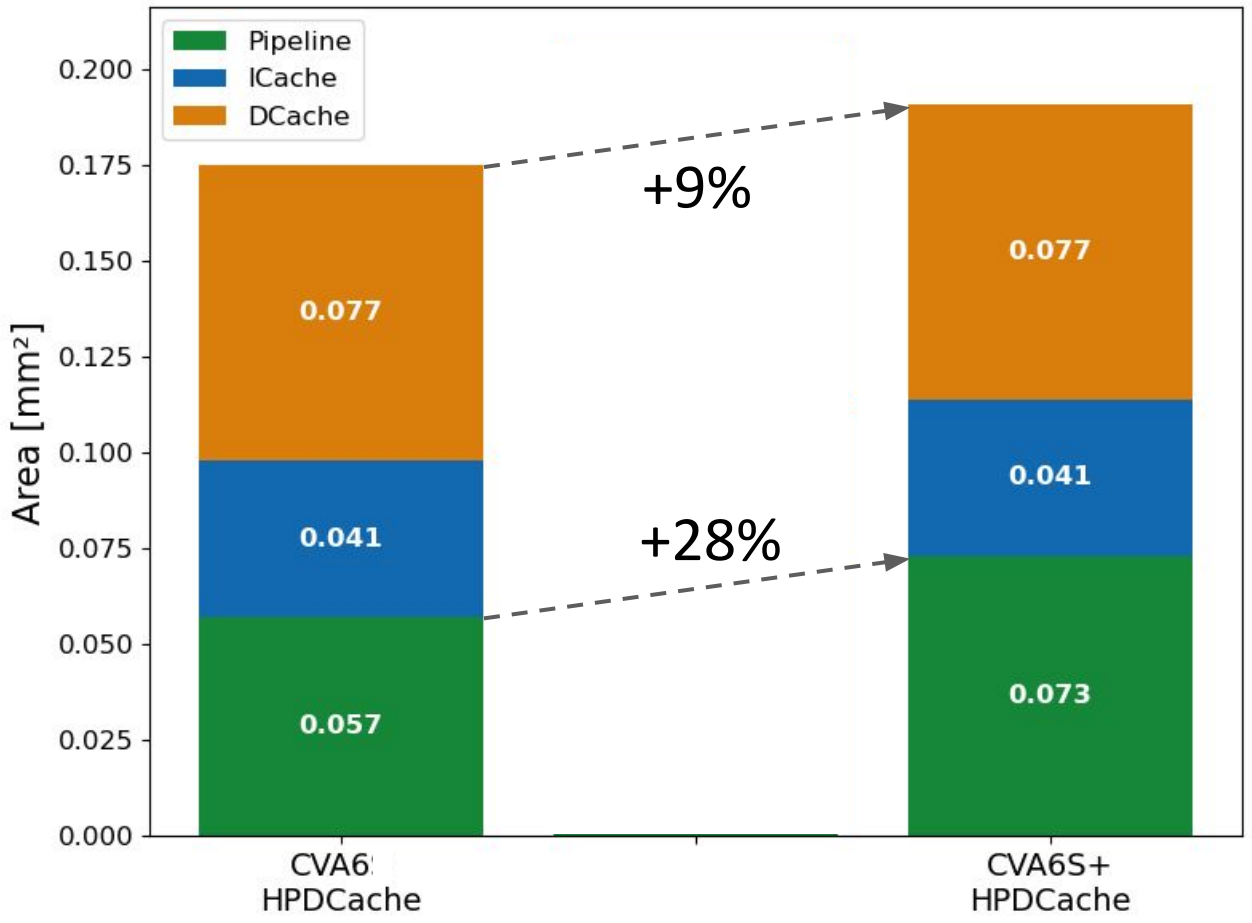
Evaluation setup:

- ➔ **GF22 nm** CMOS technology
- ➔ Worst timing corner
- ➔ **CVA6S+ versus CVA6**
- ➔ **Same caches** configuration

Pipeline area delta: +28%

Total area delta: +9%

Area and Timing: performance at what cost?

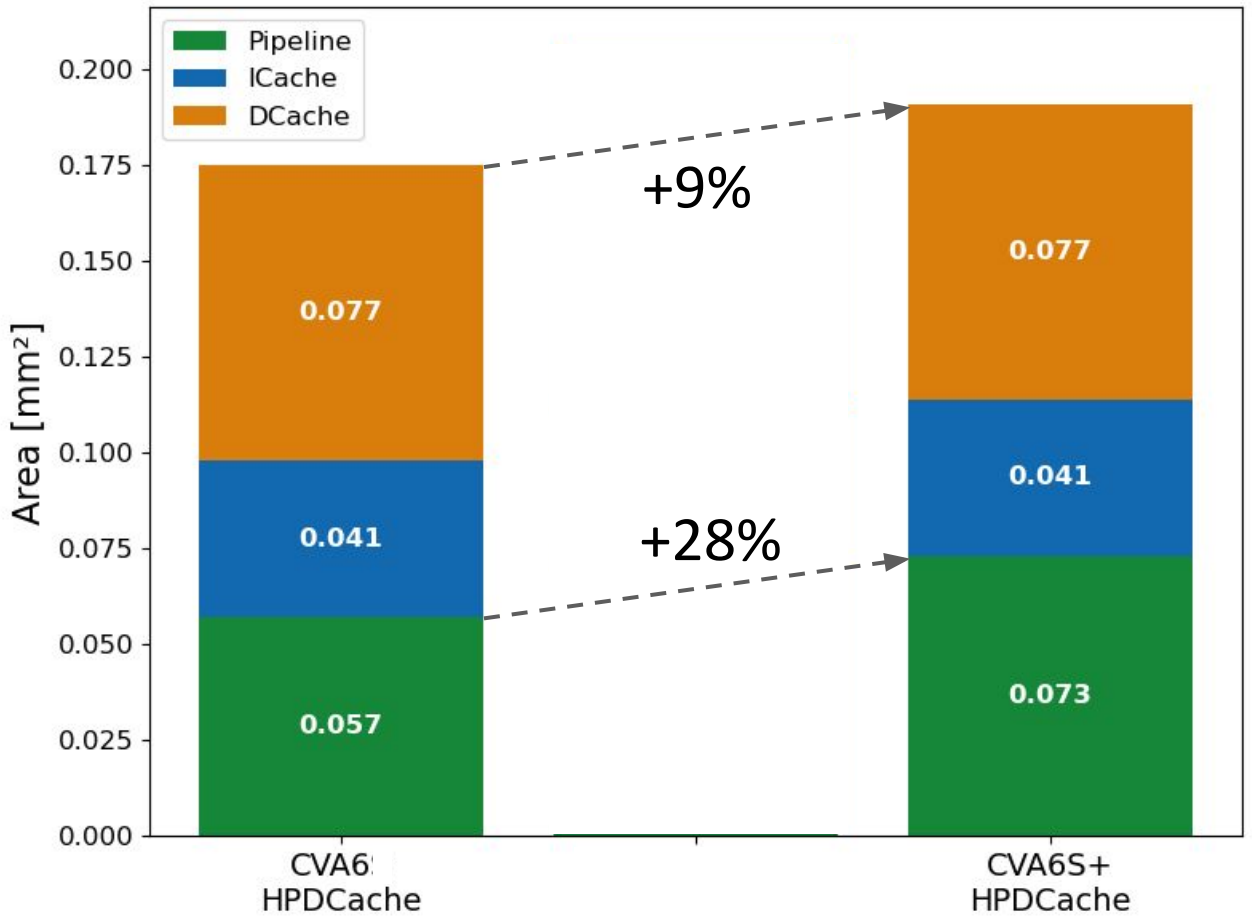


Evaluation setup:

- ➔ **GF22 nm** CMOS technology
- ➔ Worst timing corner
- ➔ **CVA6S+ versus CVA6**
- ➔ **Same caches** configuration

Pipeline area delta: +28%
Total area delta: +9%
Max. Frequency: 1090 MHz
(-0.5% vs CVA6)

Area and Timing: performance at what cost?



Evaluation setup:

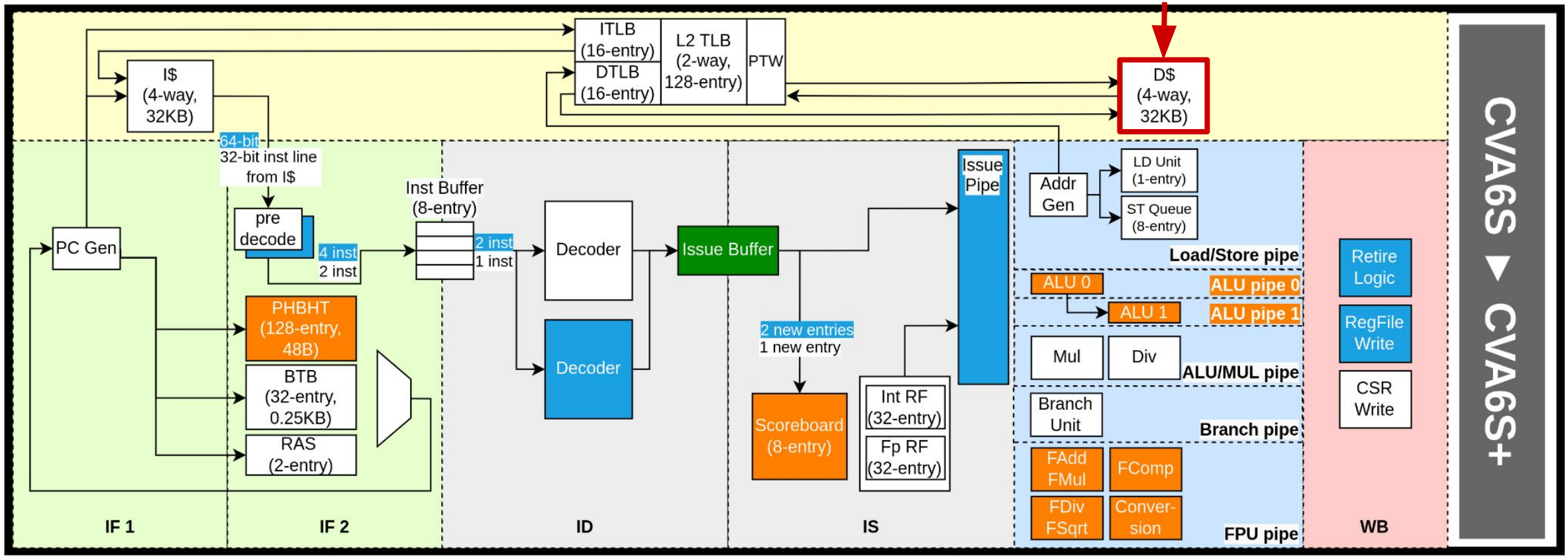
- ➔ **GF22 nm** CMOS technology
- ➔ Worst timing corner
- ➔ **CVA6S+ versus CVA6**
- ➔ **Same caches** configuration

Pipeline area delta: +28%
Total area delta: +9%
Max. Frequency: 1090 MHz
(-0.5% vs CVA6)
to obtain 43.5% IPC improvement

CVA6S+: what about the cache?

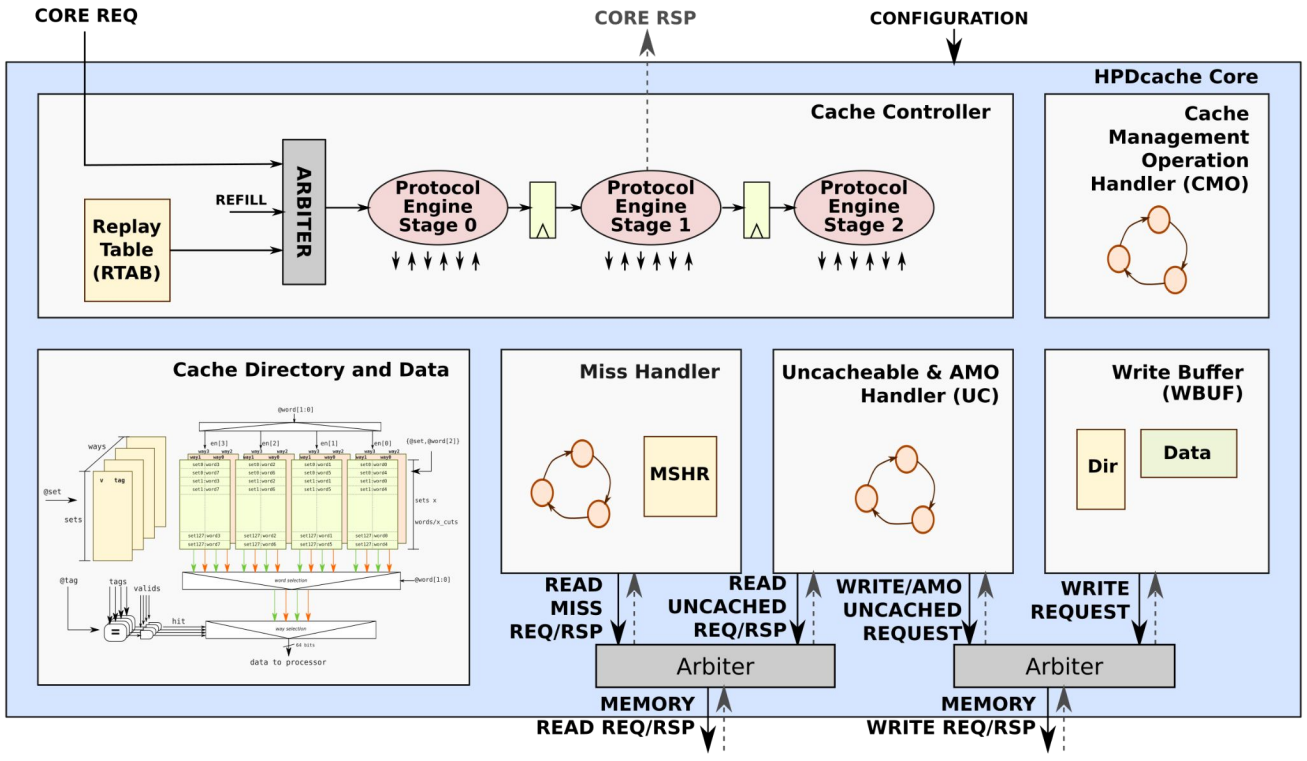
The **existing data cache subsystem** is **blocking**

Wider **Bigger** **New**



HPDCache: Open-Source High-Performance L1 D$

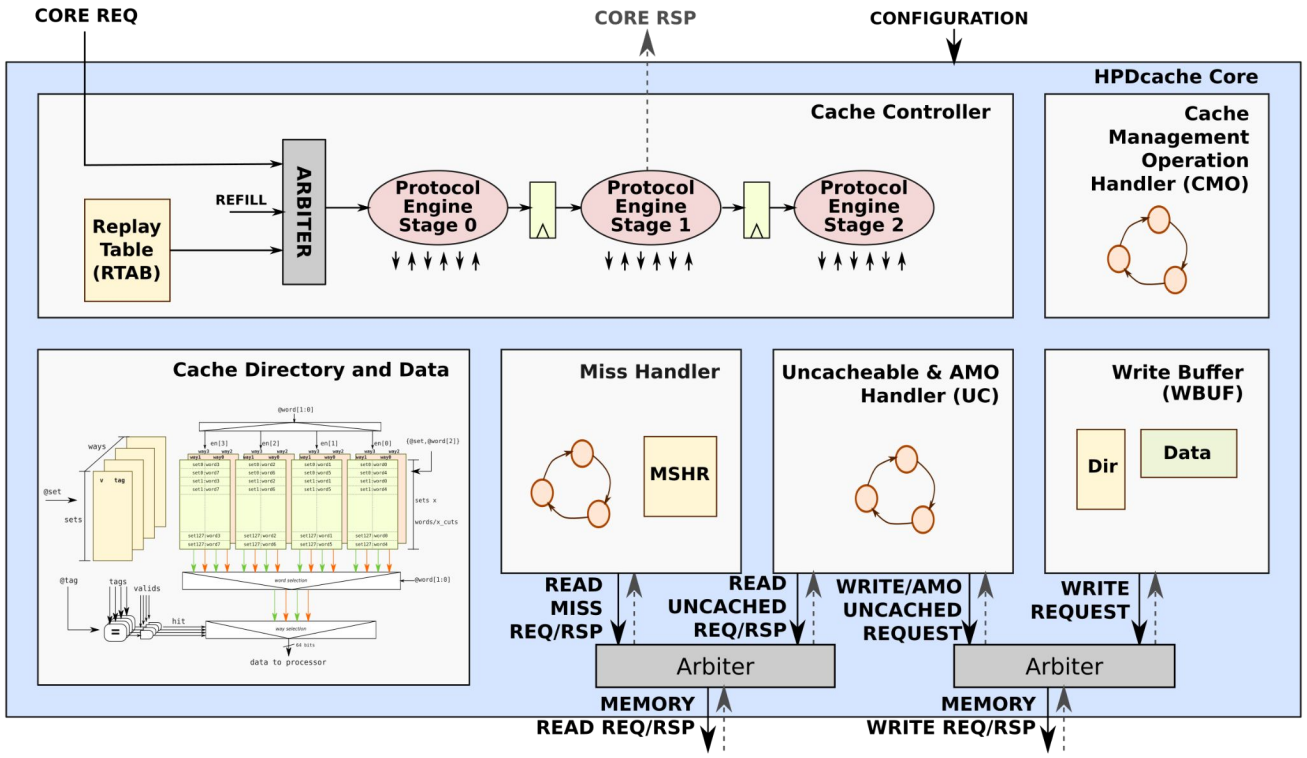
➔ **Performance-Optimized Design**: features pipelined micro-architecture, single-cycle read/write hit latency



[3]C. Fuguet, "HPDcache: Open-Source High-Performance L1 Data Cache for RISC-V Cores", ACM CF'23

HPDCache: Open-Source High-Performance L1 D$

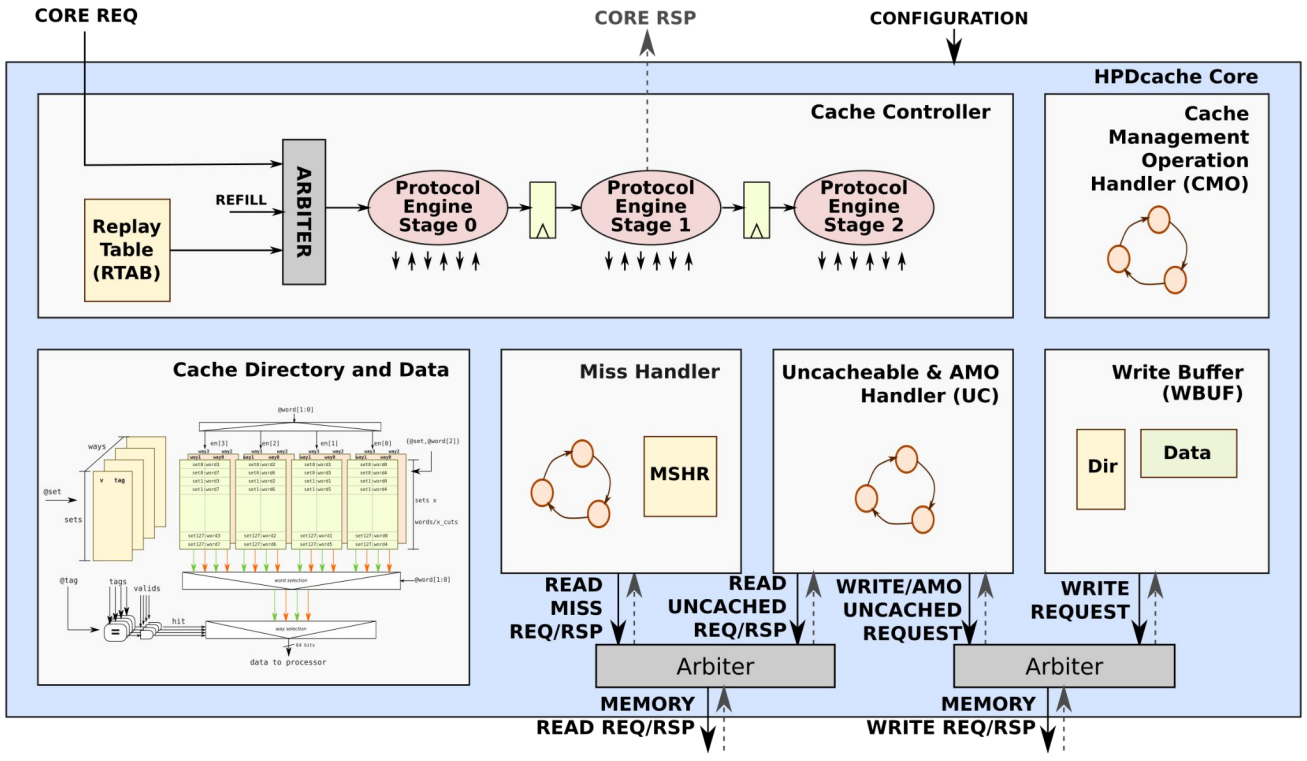
- **Performance-Optimized Design**: features pipelined micro-architecture, single-cycle read/write hit latency
- **Out-of-Order Execution & Non-Blocking**: handles requests out-of-order to avoid head-of-line blocking



[3]C. Fuguet, "HPDcache: Open-Source High-Performance L1 Data Cache for RISC-V Cores", ACM CF'23

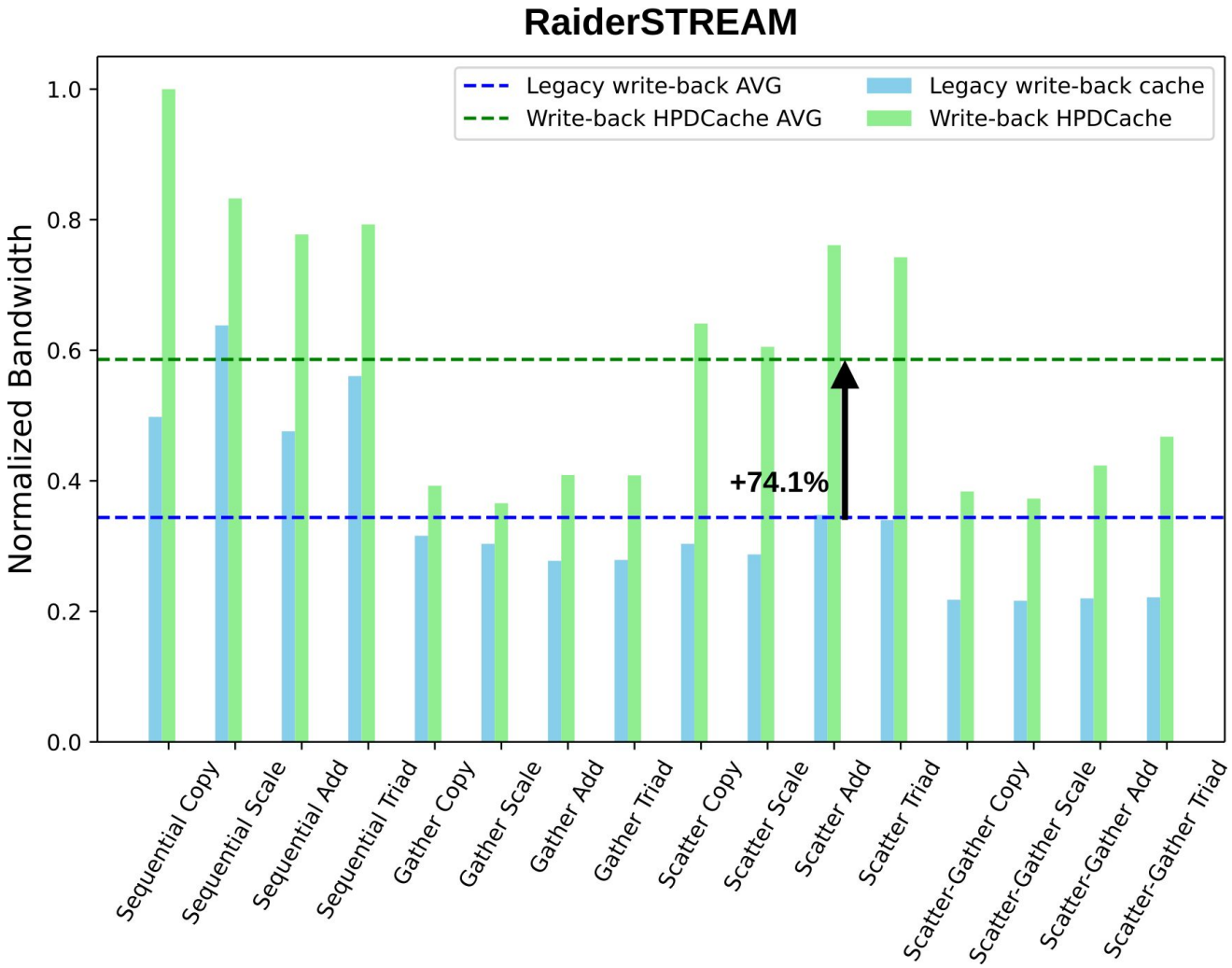
HPDCache: Open-Source High-Performance L1 D$

- **Performance-Optimized Design**: features pipelined micro-architecture, single-cycle read/write hit latency
- **Out-of-Order Execution & Non-Blocking**: handles requests out-of-order to avoid head-of-line blocking
- **Highly Configurable Architecture**: supports both WB and WT policies on a cache line-level granularity, includes configurable associativity, request port count and data widths



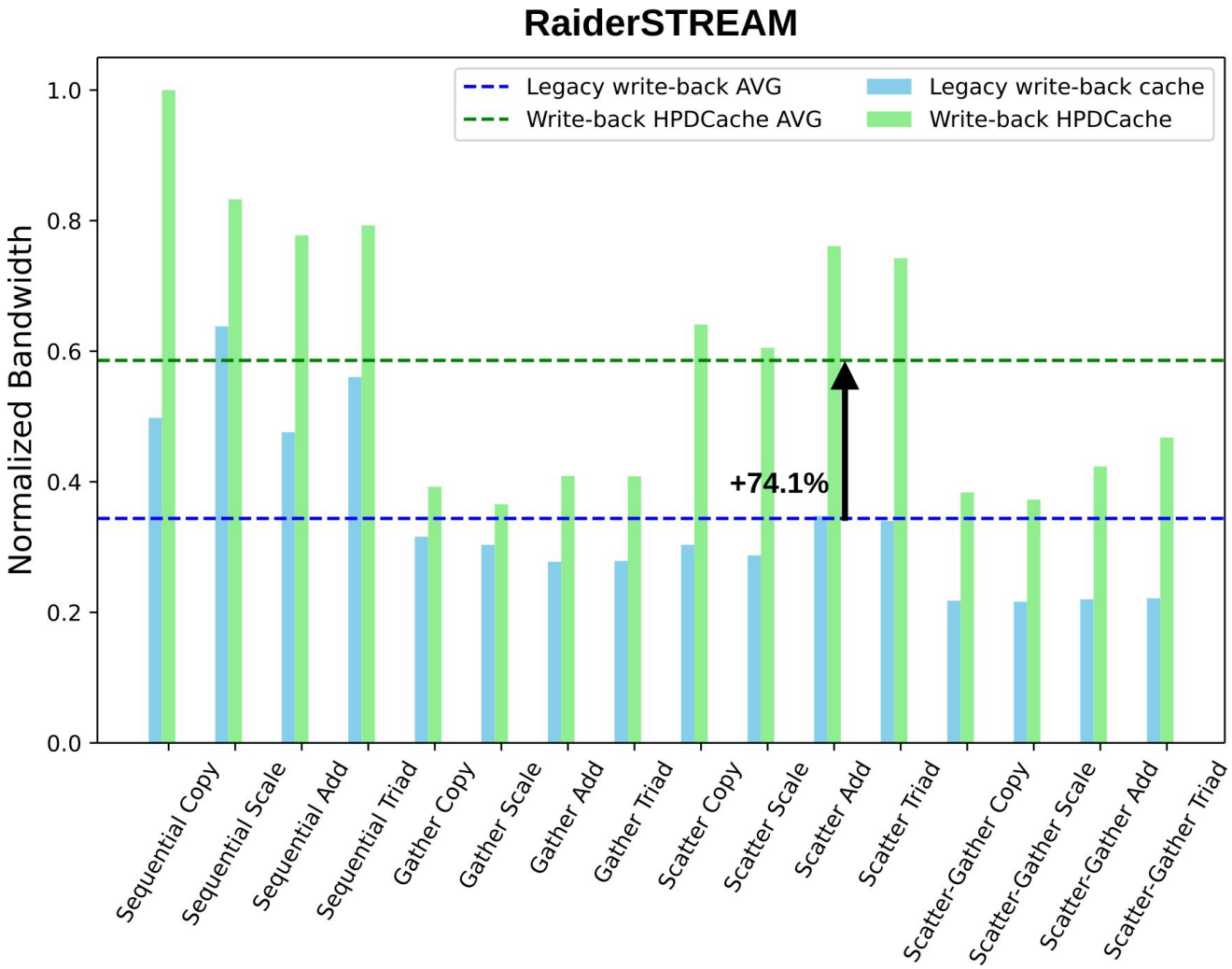
[3]C. Fuguet, "HPDcache: Open-Source High-Performance L1 Data Cache for RISC-V Cores", ACM CF'23

Cache performance: RaiderSTREAM



The **RaiderSTREAM suite** focuses on the **cache subsystem:**

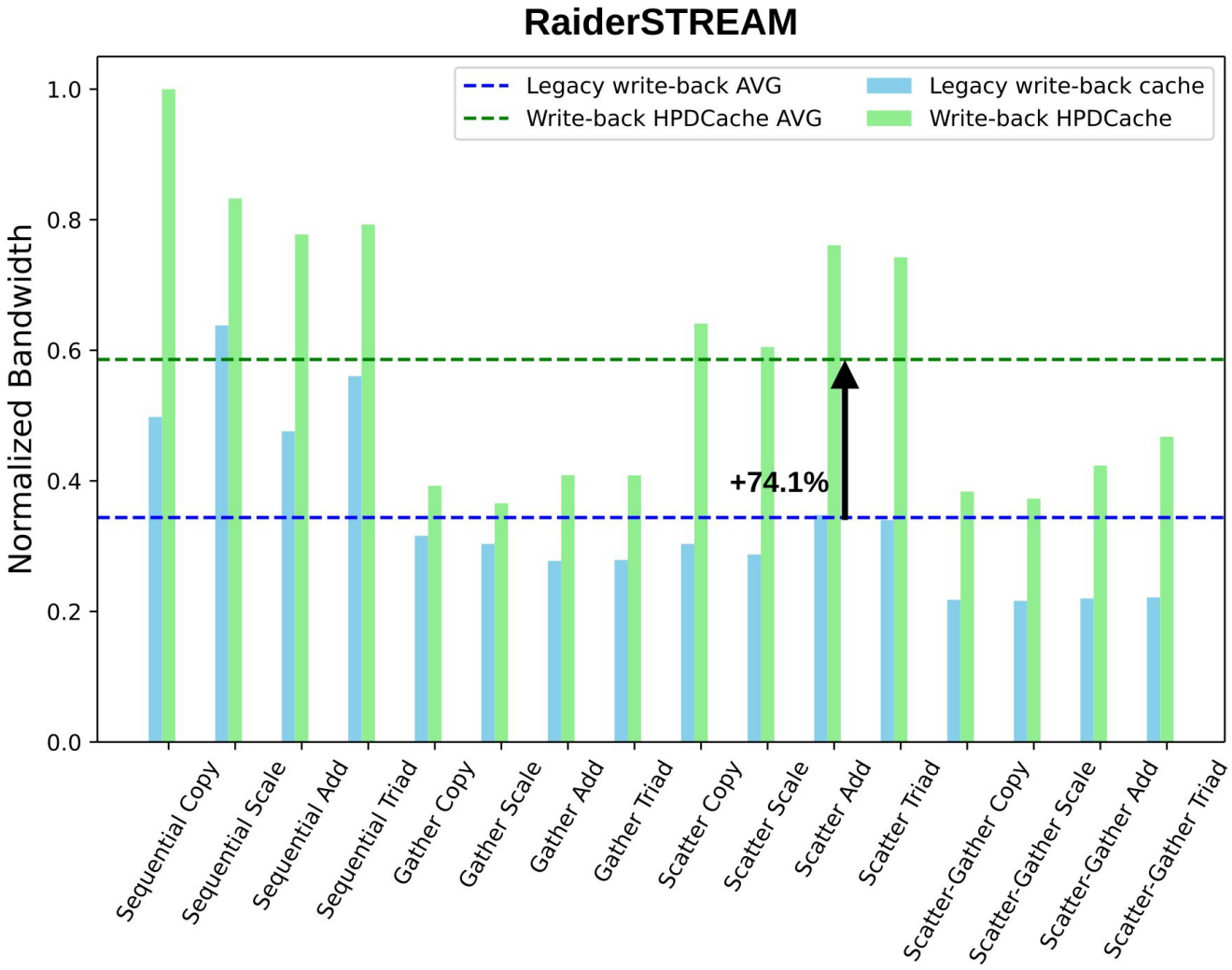
Cache performance: RaiderSTREAM



The **RaiderSTREAM suite** focuses on the **cache subsystem:**

➔ The same CVA6S+ pipeline is tested with the **legacy D$** and the **HPDCache**

Cache performance: RaiderSTREAM

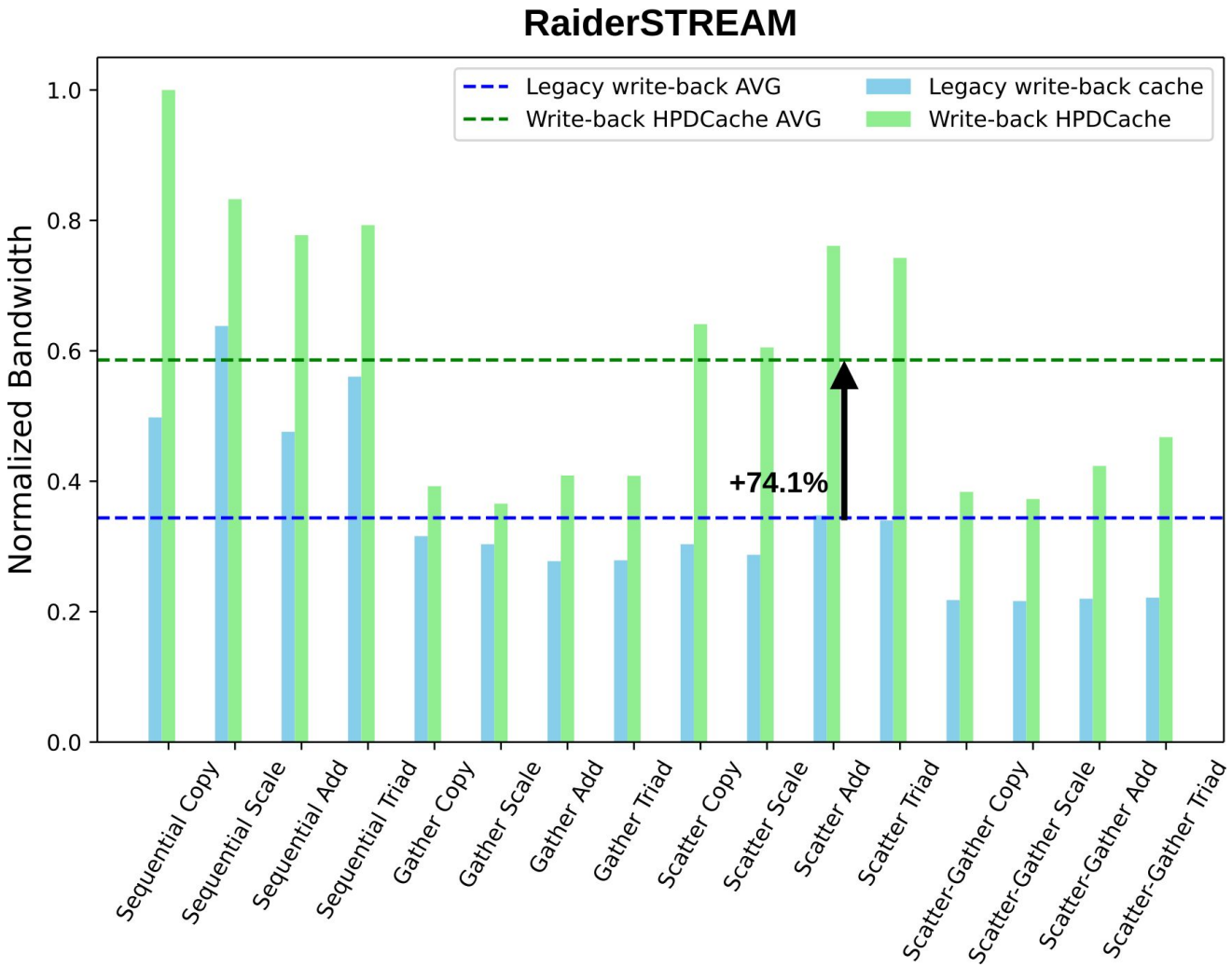


The **RaiderSTREAM suite** focuses on the **cache subsystem:**

➔ The same CVA6S+ pipeline is tested with the **legacy D$** and the **HPDCache**

➔ The **working set** is **2×** the cache capacity

Cache performance: RaiderSTREAM



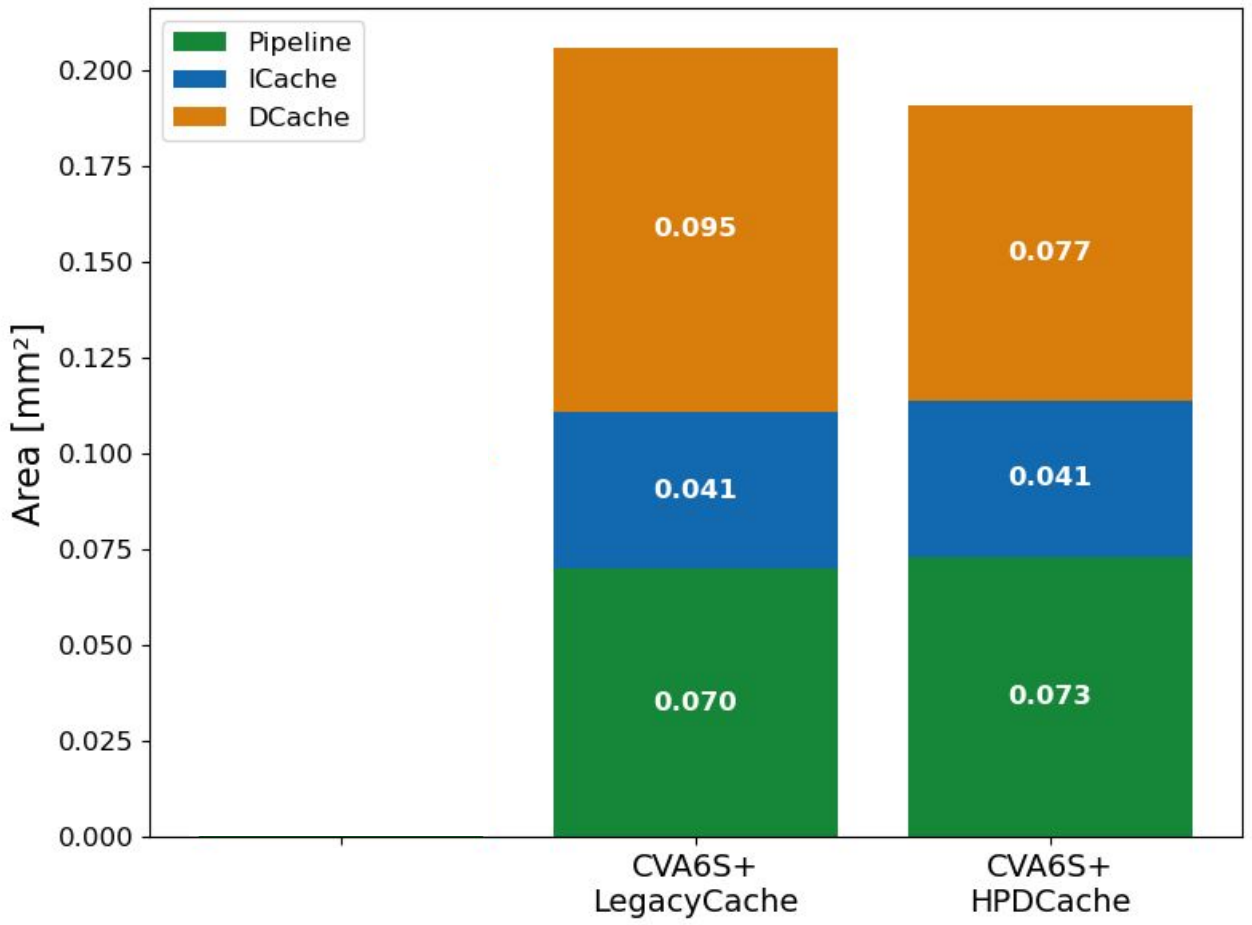
The **RaiderSTREAM suite** focuses on the **cache subsystem:**

➔ The same CVA6S+ pipeline is tested with the **legacy D$** and the **HPDCache**

➔ The **working set** is **2×** the cache capacity

+74.1% bandwidth by replacing the legacy D$ with the **HPDCache**

Area and Timing: HPDCache versus Legacy Cache

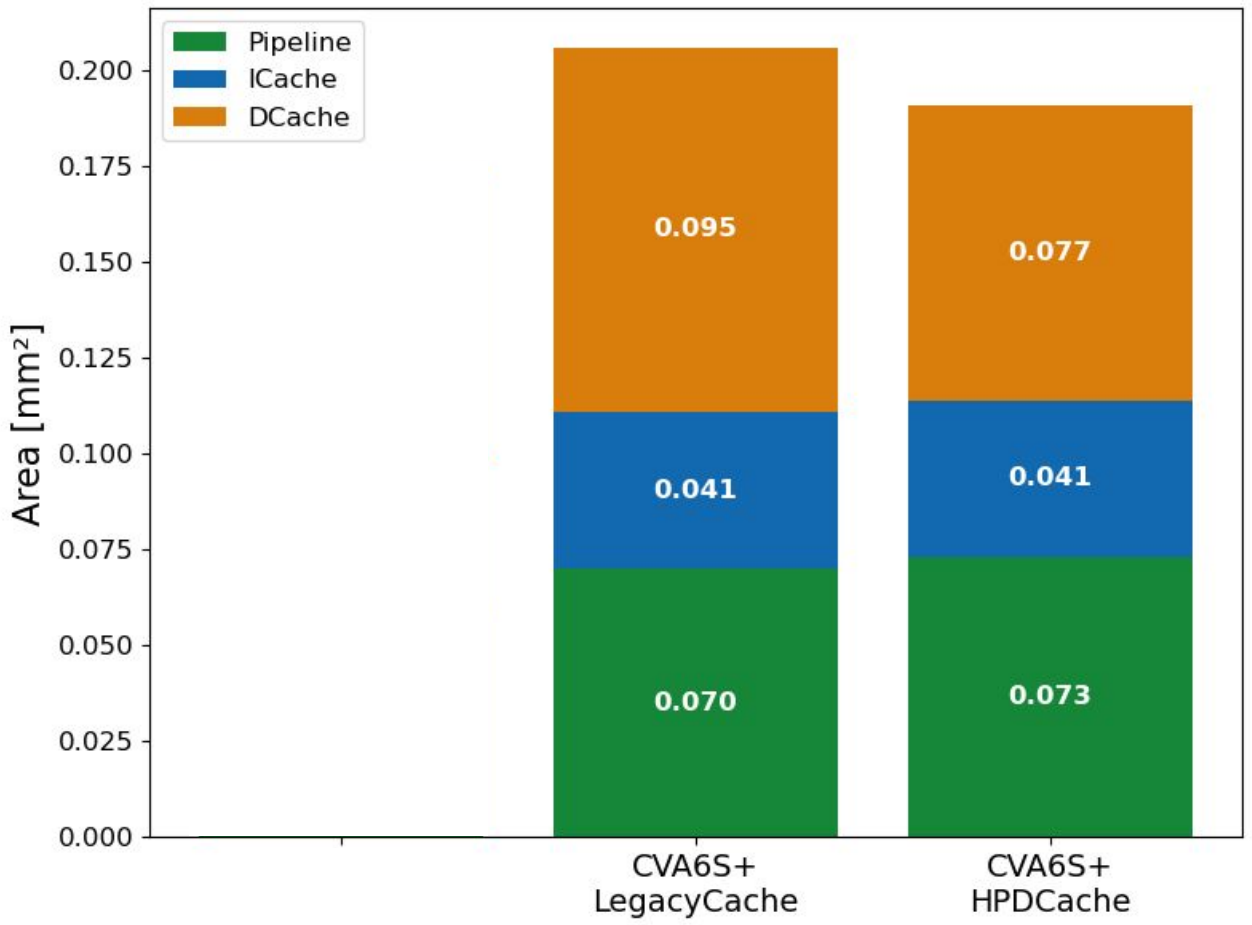


Evaluation setup:

➔ **GF22 nm** CMOS technology

➔ Worst timing corner

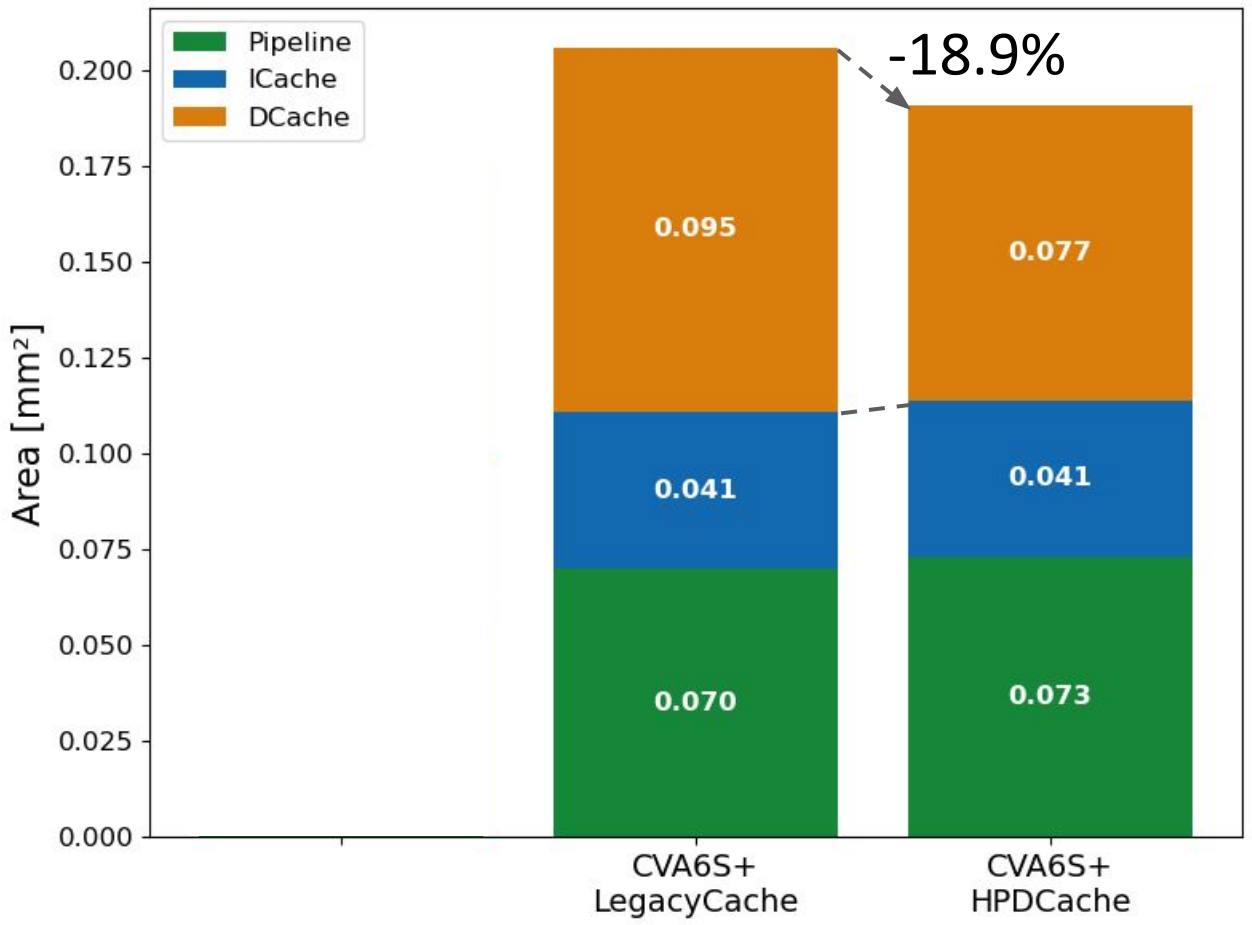
Area and Timing: HPDCache versus Legacy Cache



Evaluation setup:

- ➔ **GF22 nm** CMOS technology
- ➔ Worst timing corner
- ➔ **Legacy Cache versus HPDCache**
- ➔ **Same pipeline** configuration

Area and Timing: HPDCache versus Legacy Cache

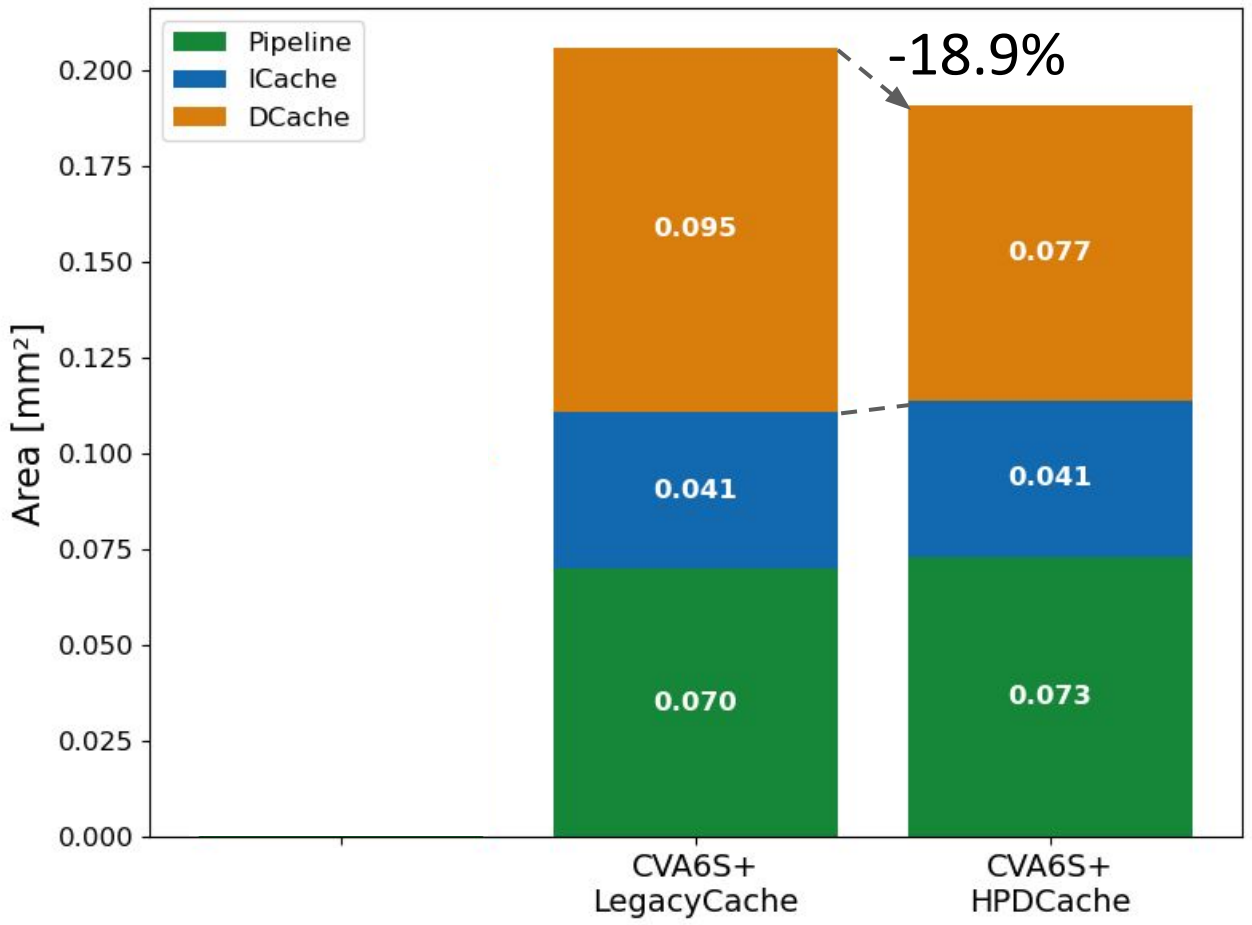


Evaluation setup:

- ➔ **GF22 nm** CMOS technology
- ➔ Worst timing corner
- ➔ **Legacy Cache versus HPDCache**
- ➔ **Same pipeline** configuration

Cache area reduction: -18.9%
due to <u>better SRAM organization</u>

Area and Timing: HPDCache versus Legacy Cache



Evaluation setup:

- ➔ **GF22 nm** CMOS technology
- ➔ Worst timing corner
- ➔ **Legacy Cache versus HPDCache**
- ➔ **Same pipeline** configuration

Cache area reduction: -18.9%
due to better SRAM organization
while providing +74.1% bandwidth!

Conclusion

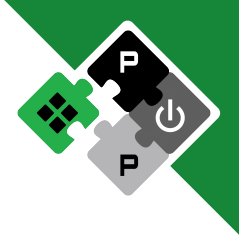
➔ We introduce **CVA6S+**, adding **key features upon CVA6S**, the superscalar dual-issue extension of the CVA6 RISC-V application-class core CVA6

Conclusion

➔ We introduce **CVA6S+**, adding **key features upon CVA6S**, the superscalar dual-issue extension of the CVA6 RISC-V application-class core CVA6

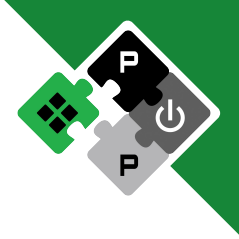
➔ We integrate **CVA6S+** with the OpenHW Core-V High-Performance L1 Data Cache **HPDCache**

Conclusion



- ➔ We introduce **CVA6S+**, adding **key features upon CVA6S**, the superscalar dual-issue extension of the CVA6 RISC-V application-class core CVA6
- ➔ We integrate **CVA6S+** with the OpenHW Core-V High-Performance L1 Data Cache **HPDCache**
- ➔ We demonstrate **10.9% and 43.5% IPC improvement over CVA6S and CVA6**, respectively, with an **area overhead of less than 10%** and **only 0.5% maximum frequency regression**

Conclusion



- ➔ We introduce **CVA6S+**, adding **key features upon CVA6S**, the superscalar dual-issue extension of the CVA6 RISC-V application-class core CVA6
- ➔ We integrate **CVA6S+** with the OpenHW Core-V High-Performance L1 Data Cache **HPDCache**
- ➔ We demonstrate **10.9% and 43.5% IPC improvement over CVA6S and CVA6**, respectively, with an **area overhead of less than 10%** and **only 0.5% maximum frequency regression**
- ➔ We showcase the **benefit of adopting the HPDCache**, which **improves the bandwidth by 74.1%** and **reduces the cache area by 18.9%**

Thank you!
Questions?

Institut für Integrierte Systeme – ETH Zürich
Gloriastrasse 35
Zürich, Switzerland

DEI – Università di Bologna
Viale del Risorgimento 2
Bologna, Italy