Architecture Support for Disciplined Approximate Programming

Hadi Esmaeilzadeh, Adrian Sampson, Luis Ceze, Doug Burger
University of Washington, Microsoft Research

Presented by: **Lucy Jiang**, **Cristina Garbacea**

Outline

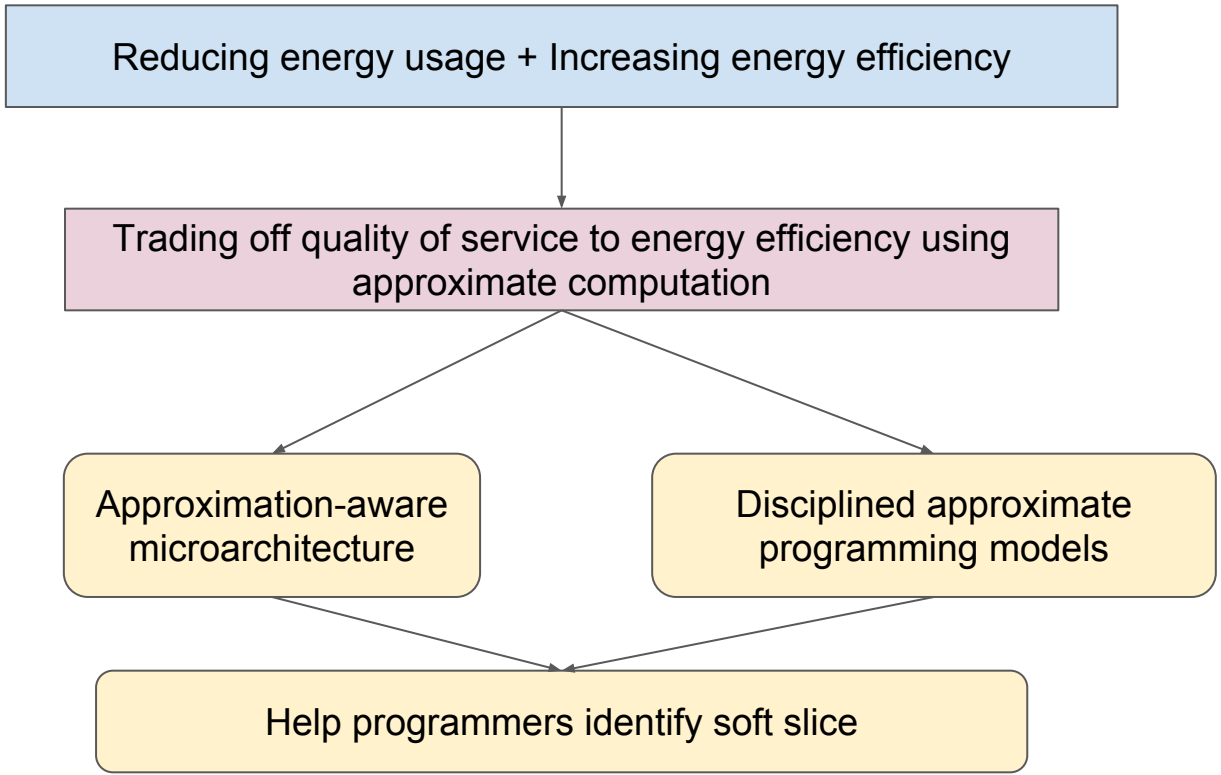
- Background
- Truffle: A Dual-Voltage Microarchitecture for Disciplined Approximation
- Evaluation
- Conclusion

Background

Goal

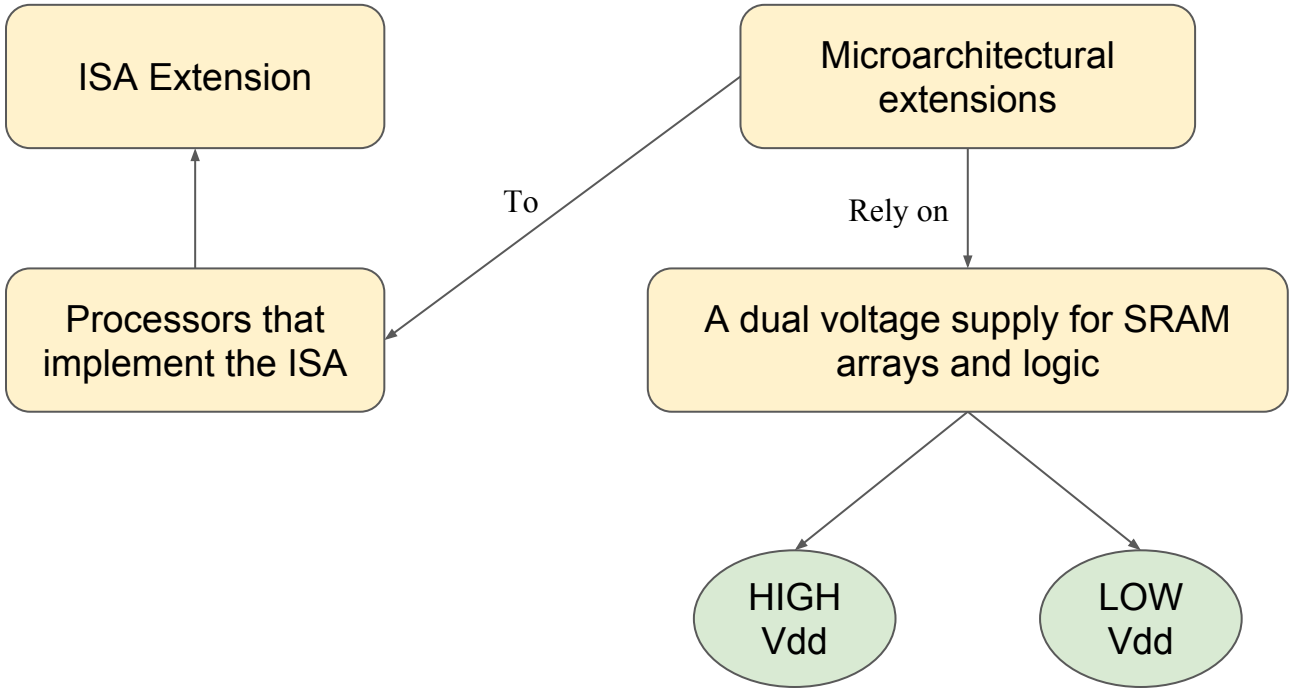
General Solution

Proposed Solution



Architecture Proposal

Allow a compiler to convey what can be approximated

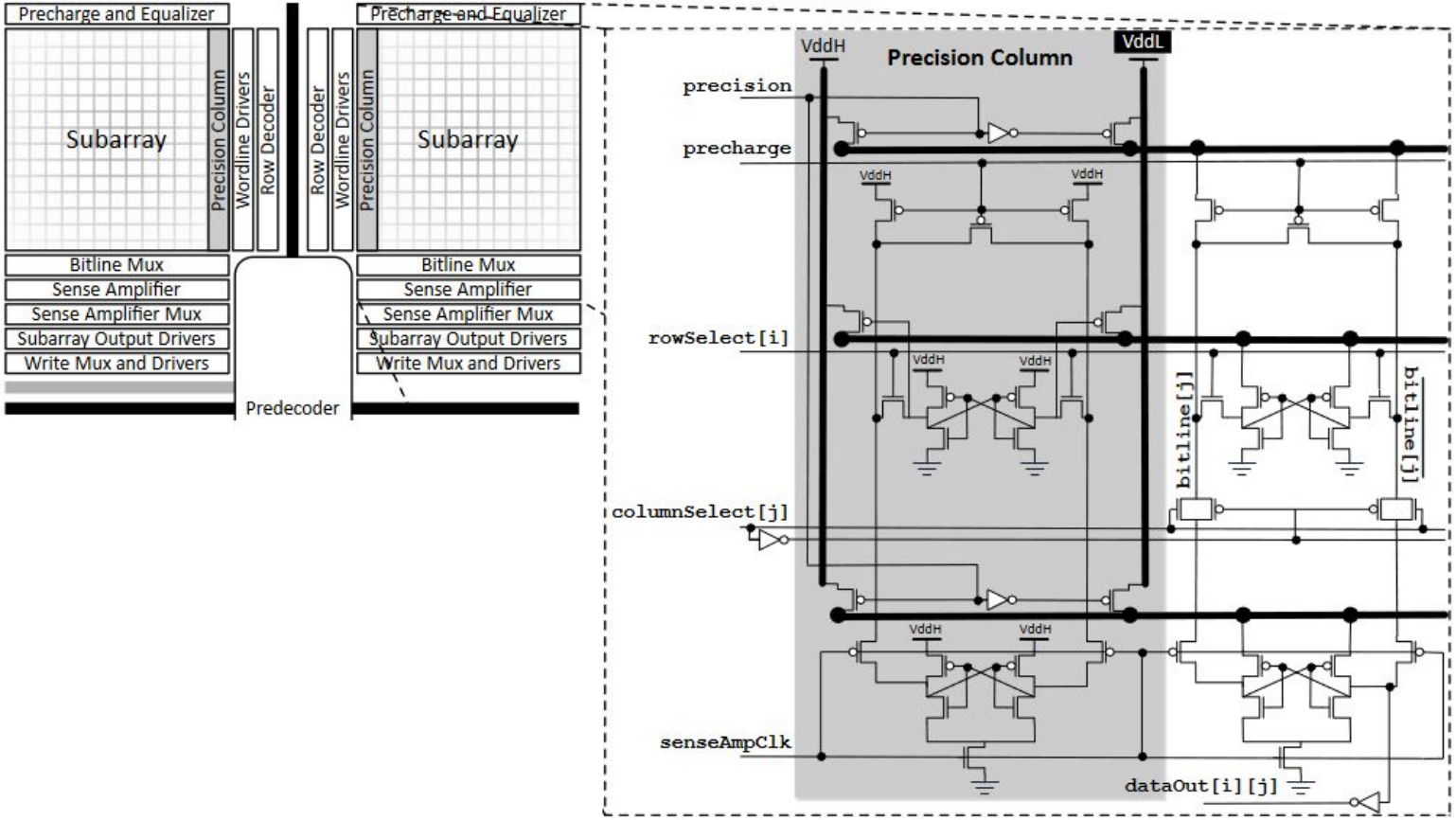


ISA Extensions for Approximation

- ISA supports both operations.
  - Precise: correct output guaranteed
  - Approximate: correct output expected
- Programming language assumption
  - Approximate doesn’t affect precise
- Precision of registers
  - Depending on the last instruction
  - Every instruction has an extra bit per operand to specify precision

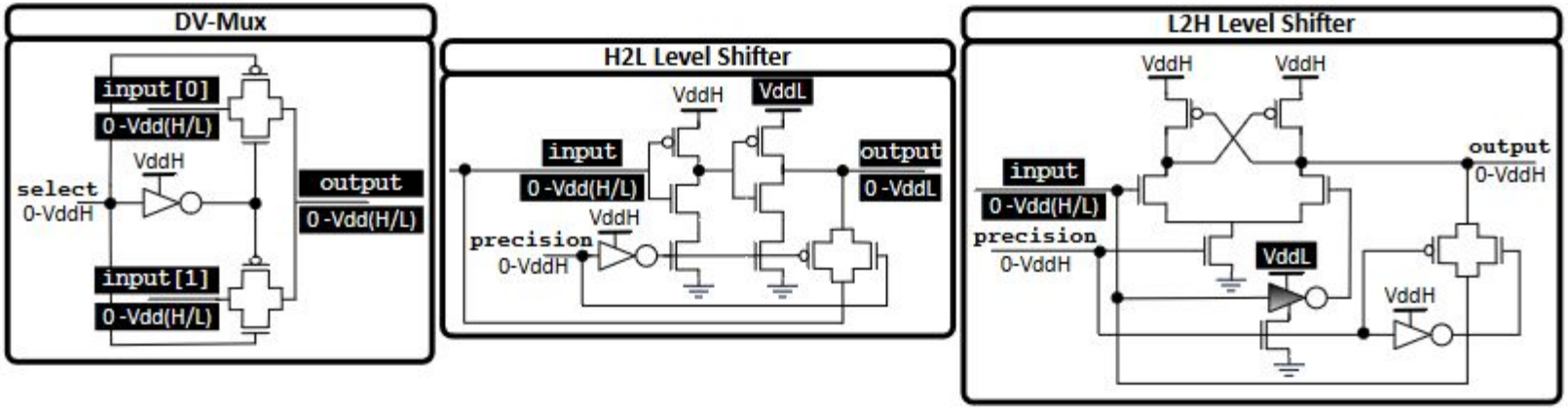
Esmaeilzadeh, Hadi, Adrian Sampson, Luis Ceze, and Doug Burger. "Architecture support for disciplined approximate programming." In ACM SIGPLAN Notices, vol. 47, no. 4, pp. 301-312. ACM, 2012.

Truffle: Dual-Voltage SRAM Structure



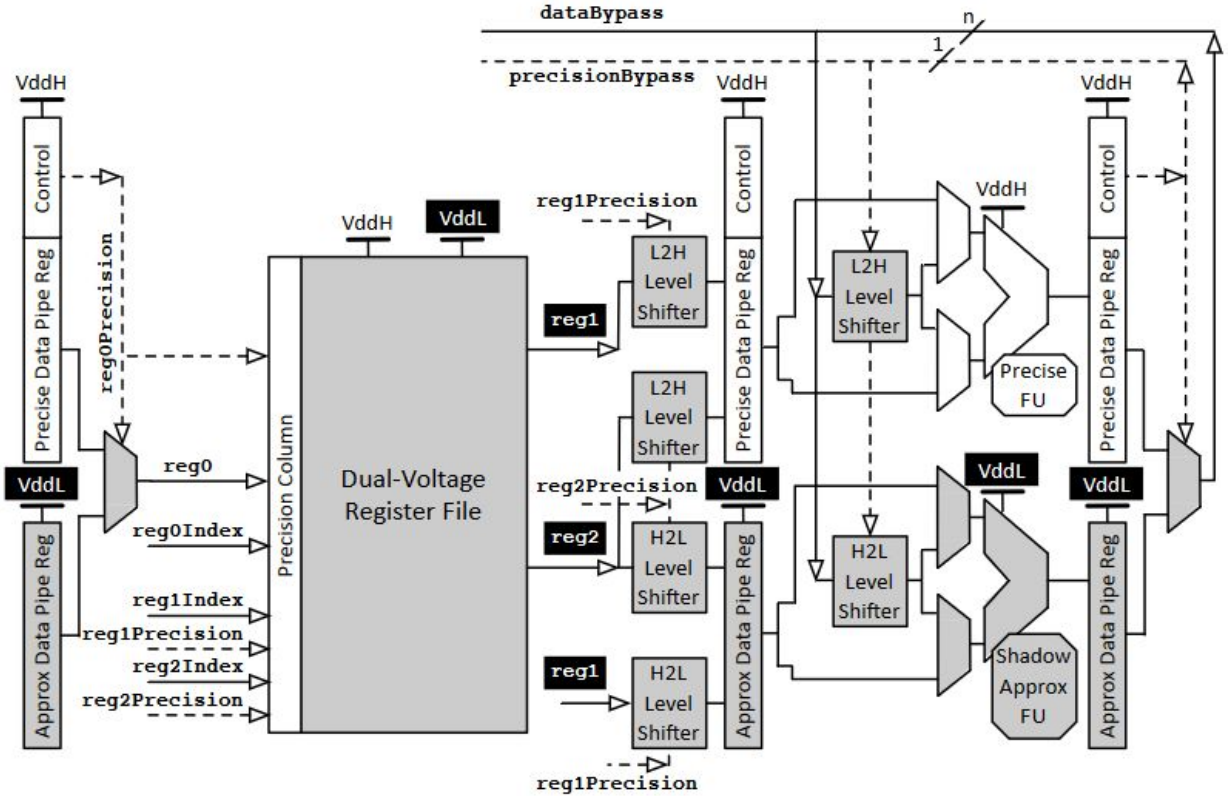
Esmaeilzadeh, Hadi, Adrian Sampson, Luis Ceze, and Doug Burger. "Architecture support for disciplined approximate programming." In ACM SIGPLAN Notices, vol. 47, no. 4, pp. 301-312. ACM, 2012.

Truffle: Voltage Level Shifting and Multiplexing



Esmaeilzadeh, Hadi, Adrian Sampson, Luis Ceze, and Doug Burger. "Architecture support for disciplined approximate programming." In ACM SIGPLAN Notices, vol. 47, no. 4, pp. 301-312. ACM, 2012.

Truffle Execution



Evaluating Truffle

Goals:

- Determine the energy savings brought by disciplined approximation
- Characterize where the energy goes
- Understand the QoS implications for software

Evaluation Setup

Table 2. Microarchitectural parameters.

Parameter	OOO Truffle	In-order Truffle
Fetch/Decode Width	4/4	2/2
Issue/Commit Width	6/4	—/—
INT ALUs/FPUs	4/2	1/1
INT Mult/Div Units	1	1
Approximate INT ALUs/FPUs	4/2	1/1
Approximate INT Mult/Div Units	1	1
INT/FP Issue Window Size	20/15	—
ROB Entries	80	—
INT/FP Architectural Registers	32/32	32/32
INT/FP Physical Registers	80/72	—
Load/Store Queue Size	32/32	—
ITLB	128	64
I Cache Size	64 Kbyte	16 Kbyte
Line Width/Associativity	32/2	16/4
DTLB	128	64
D Cache Size	64 Kbyte	32 Kbyte
Line Width/Associativity	16/2	16/4
Branch Predictor	Tournament	Tournament

- Truffle is modeled at 65 nm technology node in the context of both in-order and out-of-order designs

- Evaluate each benchmark for two criteria: **energy savings** and **sensitivity to error**

- Statistics collected: variable, field, array accesses, basic blocks, arithmetic and logical operators

- Inject errors in the execution and measure the consequent degradation in output quality

Evaluation Setup

Benchmarks:

- 9 benchmark programs written in EnerJ:
  hand-annotated programs with approximate type qualifiers that distinguish their approximate parts (SciMark2, ZXing, jMonkeyEngine, ImageJ, 3D raytracer)

- How to quantify the loss in output quality caused by hardware approximation?
  Define an application specific quality-of-service metric to quantify the loss in output quality
  - ❖ RMSE
  - ❖ Proportion of incorrect intersection decisions
  - ❖ Proportion of unsuccessful decodings of a sample QR code image

Energy Savings

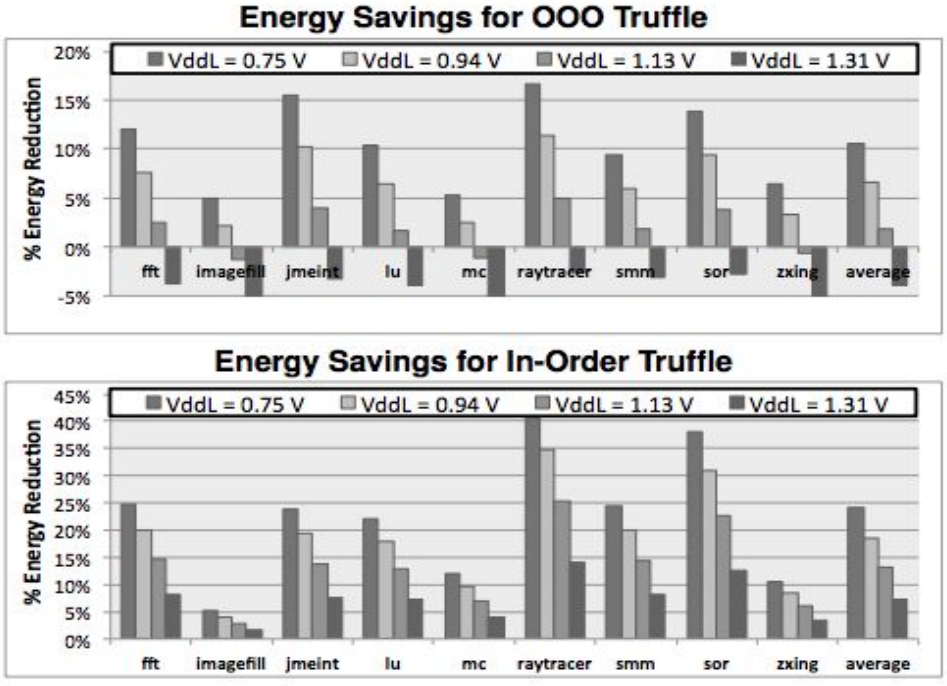


Figure 5. Percent energy reduction with unchecked OOO and in-order Truffle designs for various $V_{dd}L$ voltages.

$V_{dd}H$ = 1.5 V

$V_{dd}L$: 50%, 62.5%, 75%, 87.5% of $V_{dd}H$

Frequency = 1666 MHz

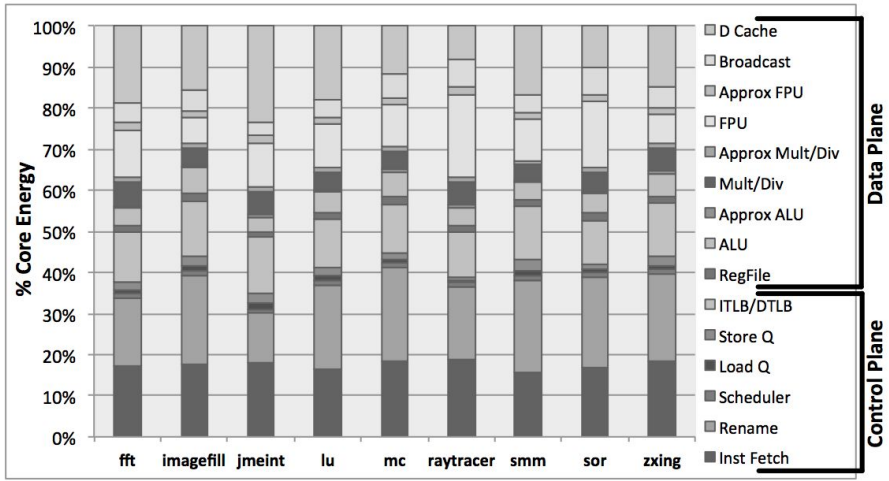
- in-order configuration: all voltage levels lead to energy savings (up to 43%)

- out-of-order configuration: energy savings when **$V_{dd}L$** is less than 75% of **$V_{dd}H$**

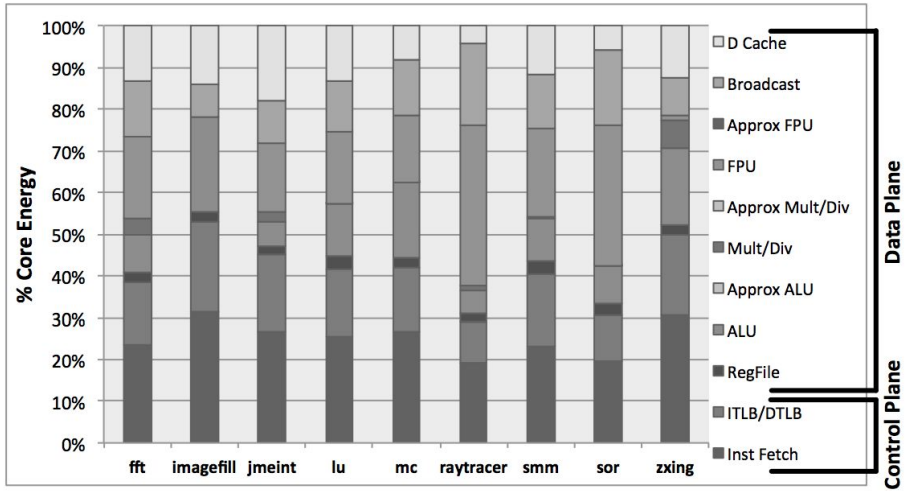
The impact of Truffle in in-order cores is much higher!

Energy Breakdown Per Component

Out-Of-Order Truffle



In-Order Truffle



Imagefill: 42% (OOO) and 47% (In-Order) of energy is consumed in data movement/processing plane
Raytracer: 71% (OOO) and 50% (In-Order)

Energy Savings Potential

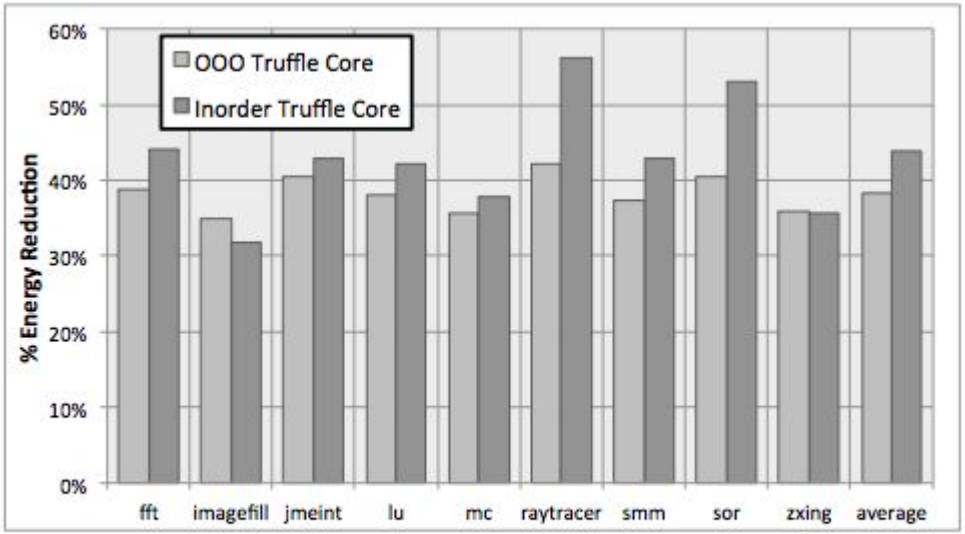


Figure 8. Percent energy reduction *potential* for checked in-order and OOO Truffle designs with $V_{dd}L = 0.75$ V.

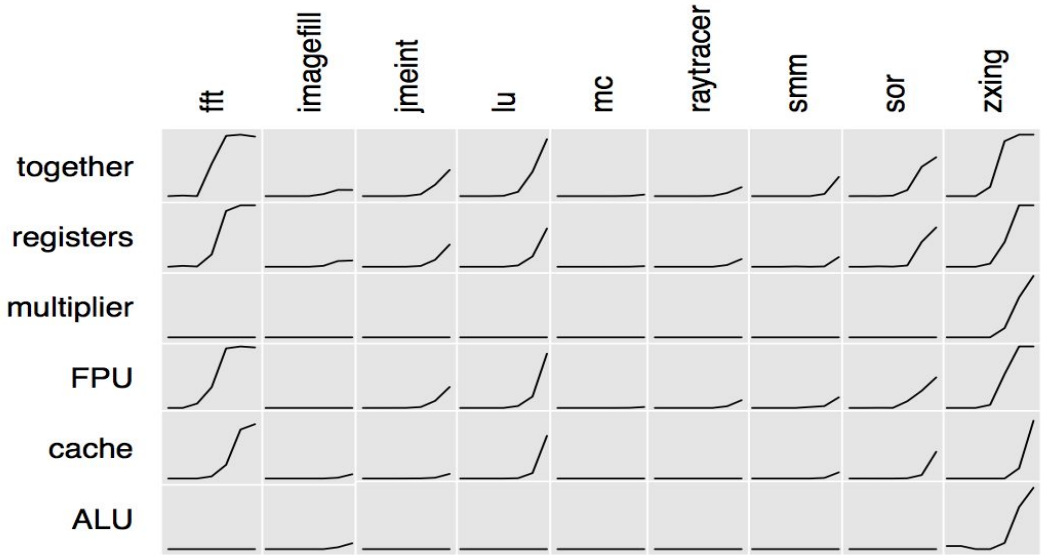
- reduce the voltage level of the instruction control plane to 1.2 V

$V_{dd}L$ = 0.75 V

- the gap in energy savings potential between the OOO and in-order designs is significantly reduced.

Error Propagation from Circuits to Applications

- Inject errors in each of the microarchitectural structures that support approximate behaviour



Conclusion

- Disciplined approximate programming is an effective and usable technique for trading off correctness guarantees with energy savings

- Dual-voltage microarchitectures can provide both approximate and precise computation controlled at a fine-grain by the compiler; ISA relies on the compiler to eliminate the need for checking or recovery at run time

- Truffle presents energy savings up to 43% under reasonable assumptions, with benchmarks exhibiting negligible degradation in output quality.

Discussion points

1. For approximation, should we rely more on hardware support or software support? Why?

2. Is Truffle practical enough to be implemented in real world applications?

3. What kind of applications are suitable for disciplined approximate programming using this microarchitecture?