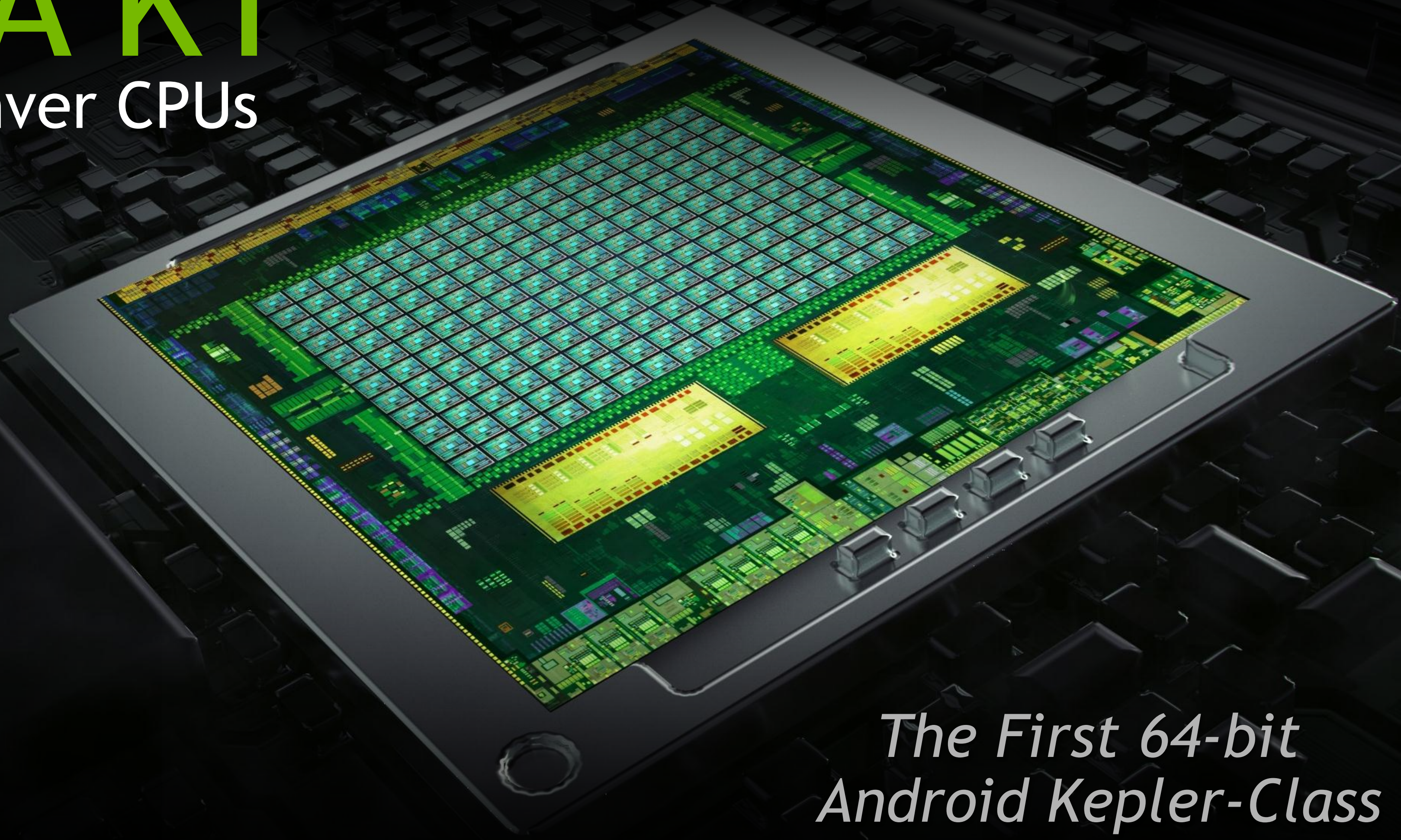
HOT CHIPS 2014
NVIDIA'S DENVER PROCESSOR

Darrell Boggs, CPU Architecture

Co-authors: Gary Brown, Bill Rozas,
Nathan Tuck, K S Venkatraman

TEGRA K1
with Dual Denver CPUs



The First 64-bit Android Kepler-Class Chip

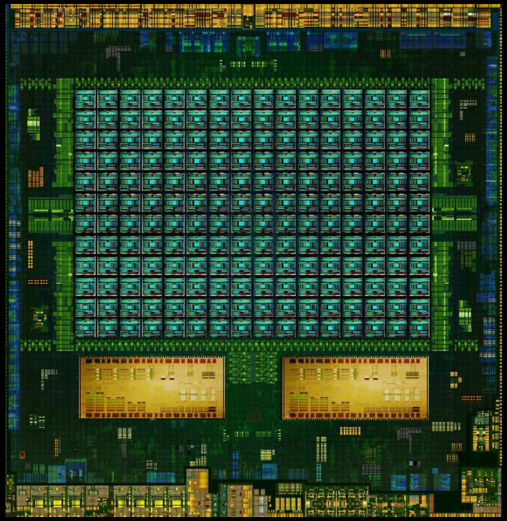
NVIDIA

TEGRA K1

192-core
Kepler-Class Chip

One Chip — Two Versions

Pin
Compatible



Quad A15 CPUs	Dual Denver CPUs
32-bit	64-bit
3-way Superscalar	7-way Superscalar
Up to 2.3GHz	Up to 2.5GHz
32K+32K L1$	128K+64K L1$

DENVER VALUE PROPOSITION

Highest performance and very energy-efficient ARMv8 processor

- Greater dynamic sharing with GPU
- Extended battery life
- Low latency power-state transition
- Best web browsing experience

Designed to bring PC-class performance to the ARM ecosystem

- Content creation
- Gaming
- Enterprise applications

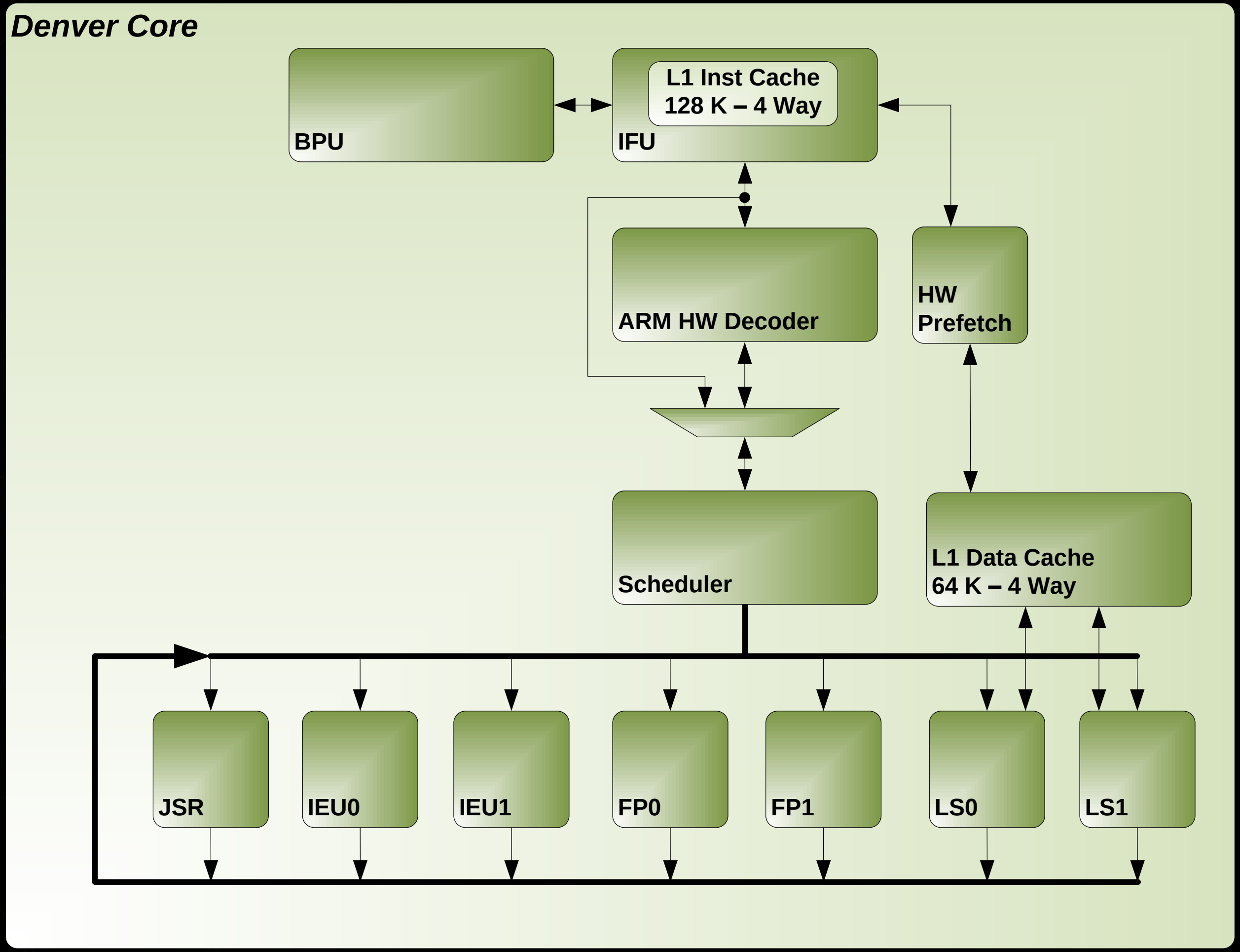
DENVER CPU

Highest Perf ARMv8 CPU

- 7-wide superscalar
- Aggressive HW prefetcher

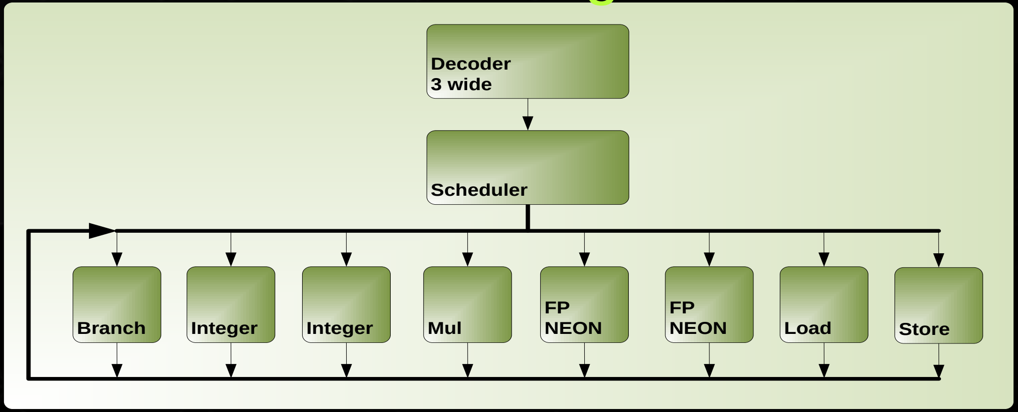
Dynamic Code Optimization

- Optimize once, use many times
- OOO execution without the power



TEGRA K1 SUPERSCALAR ARCHITECTURE

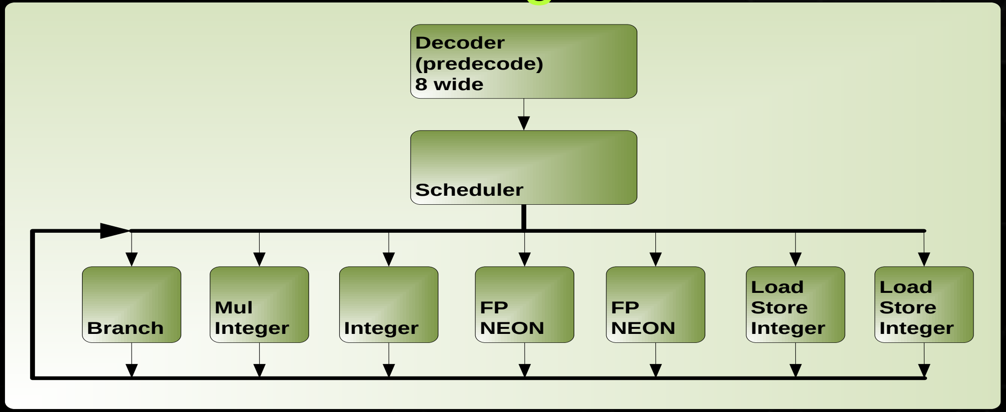
Cortex-A15 Tegra K1-32



- Branch: 1
- Integer: 2
- Multiply: 1
- Floating Point/Neon: 2 x 64-bit
- LD/ST: 1 LD and 1 ST

Peak IPC 3

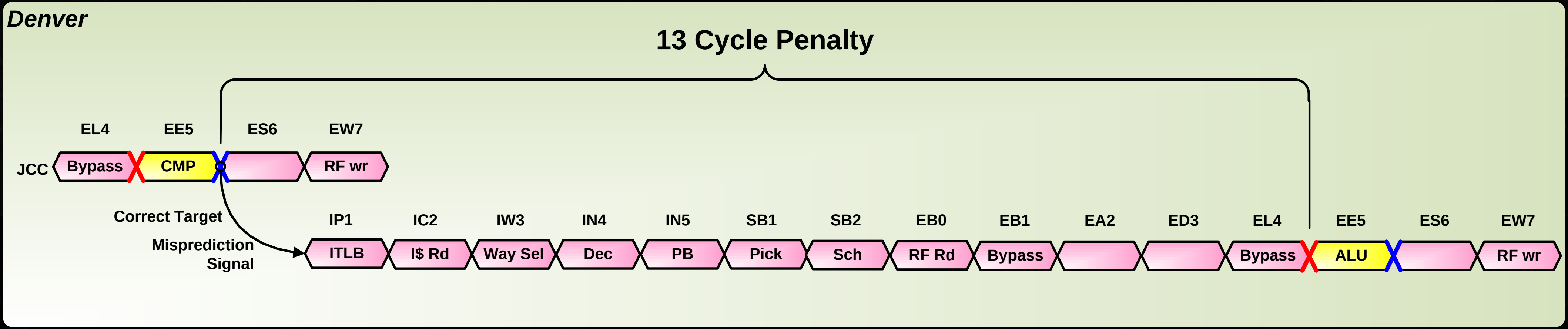
Denver Tegra K1-64

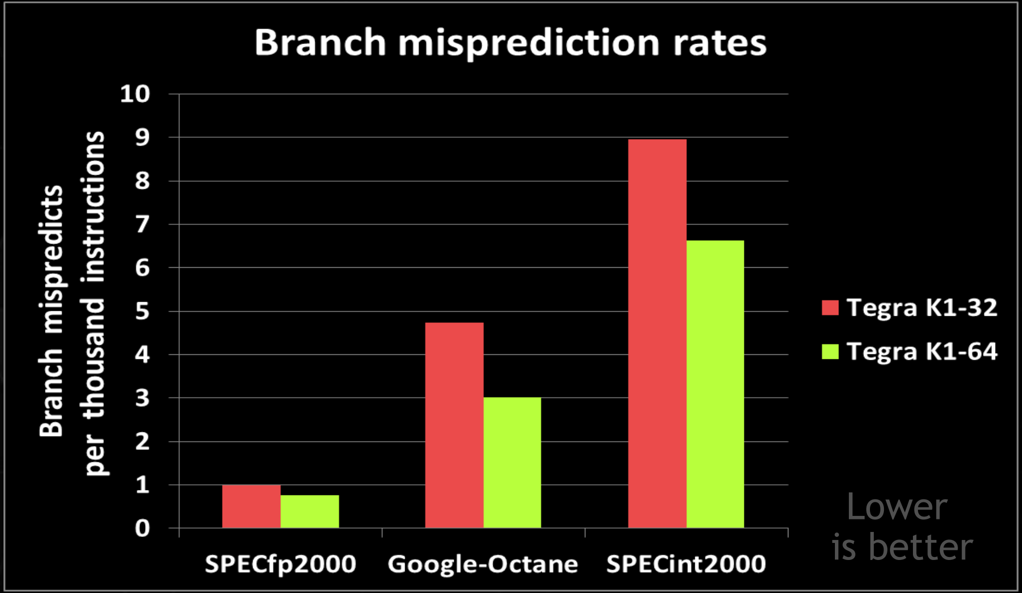


- Branch: 1
- Integer: 2 (+ Mul) + 2
- Floating Point/Neon: 2 x 128-bit
- LD/ST: 2 LD and/or ST

Peak IPC 7+

Pipeline Microarchitecture - Mispredict Penalty



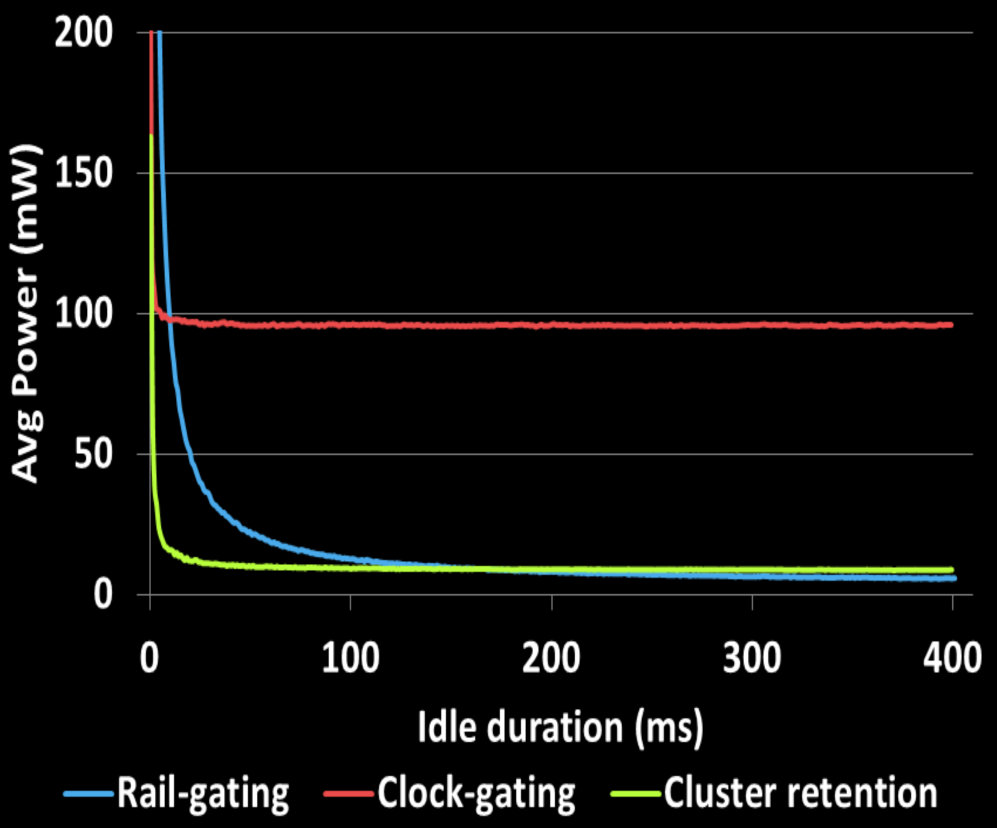


- Tegra K1-32
  - 15 cycle mispredict
- Tegra K1-64
  - 13 cycle mispredict

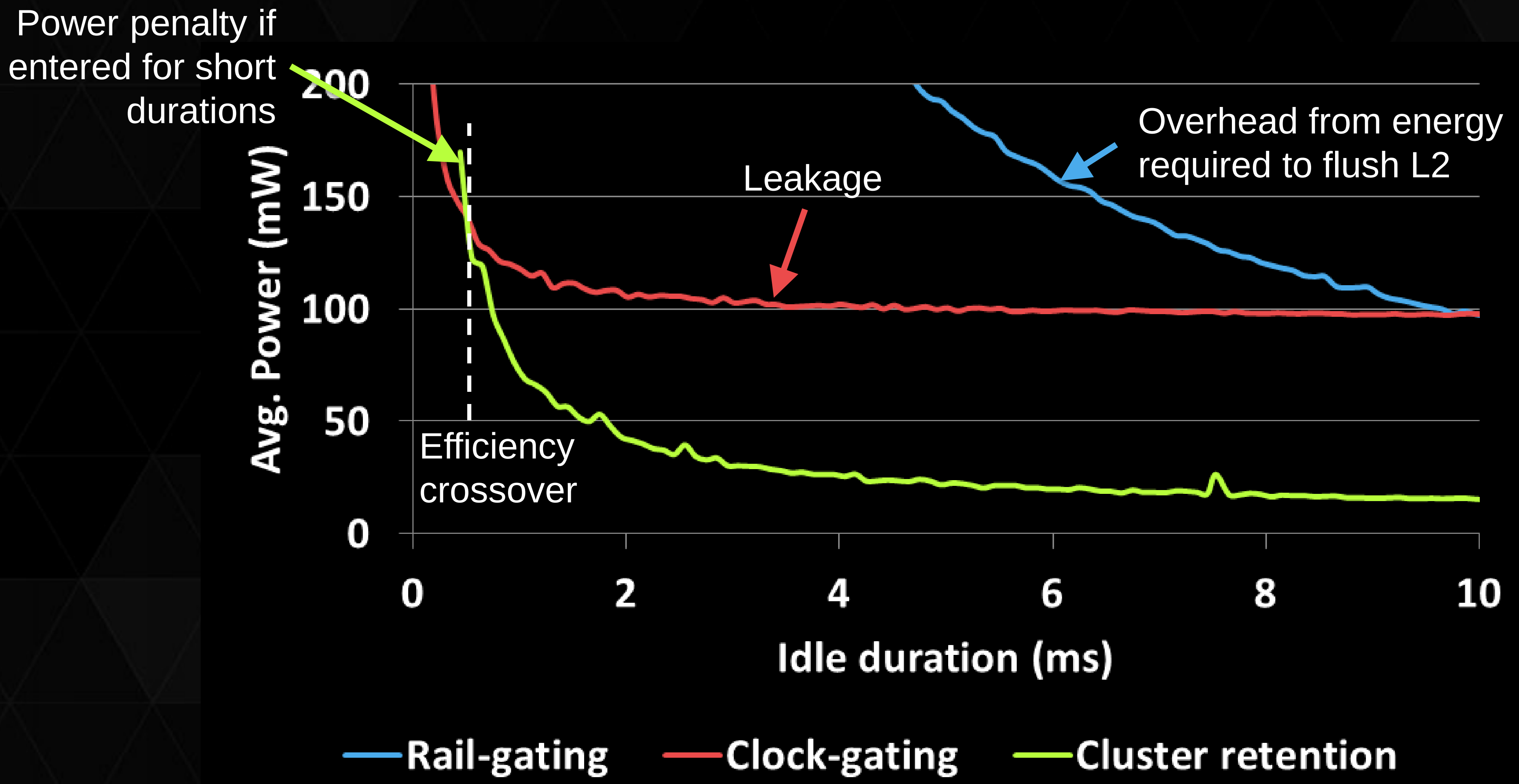
Higher ILP and efficiency

CORE CLUSTER RETENTION STATE

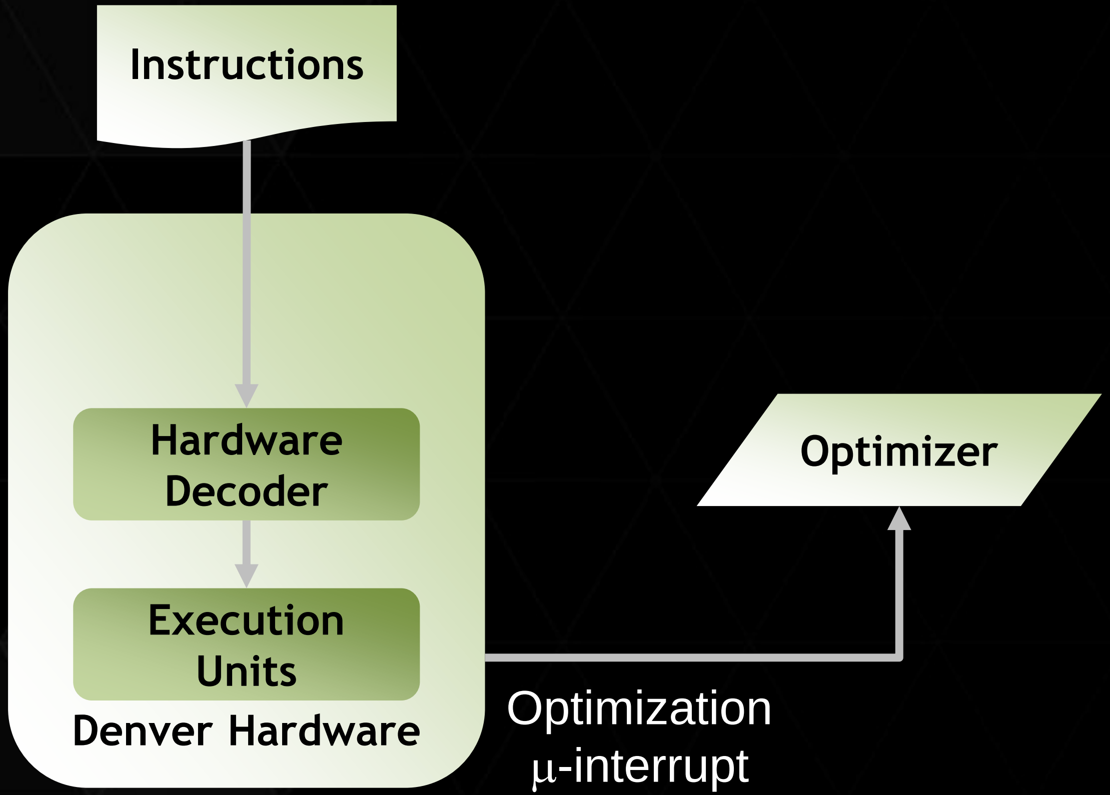
- New power management state: CC4
- Allows cache and architectural state retention
- Allows voltage to be reduced below Vmin to a retention voltage
- Fast entry and exit latencies enable wider range of use



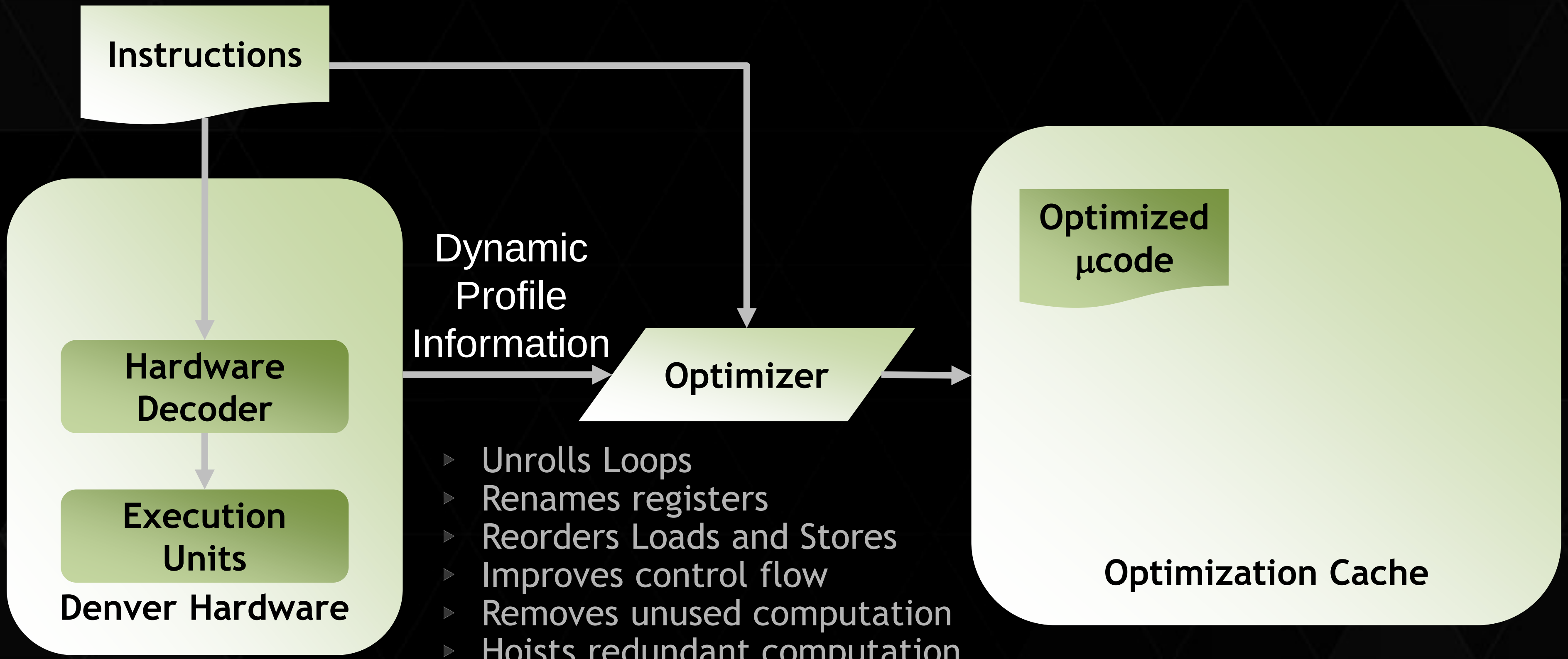
DENVER IDLE POWER IMPROVES WITH RETENTION



DYNAMIC CODE OPTIMIZATION
OPTIMIZE ONCE, USE MANY TIMES



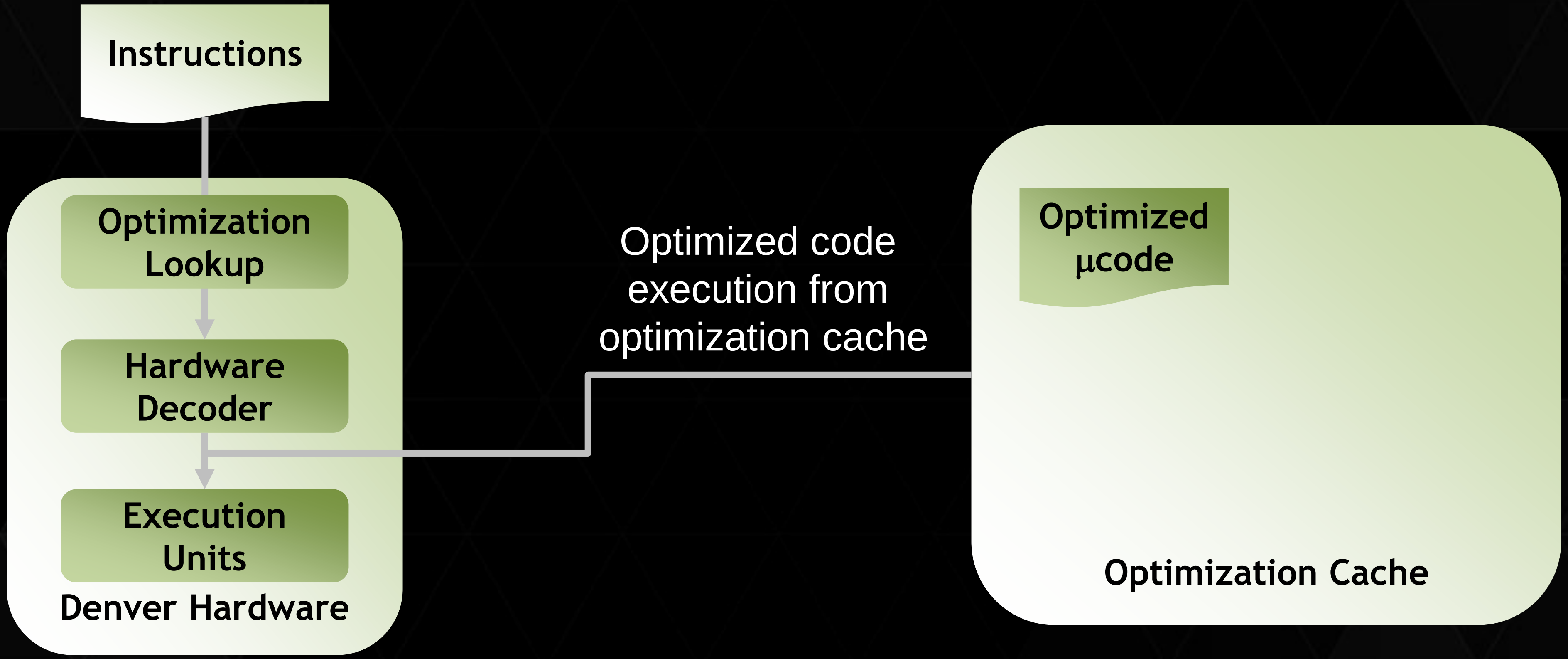
DYNAMIC CODE OPTIMIZATION
OPTIMIZE ONCE, USE MANY TIMES



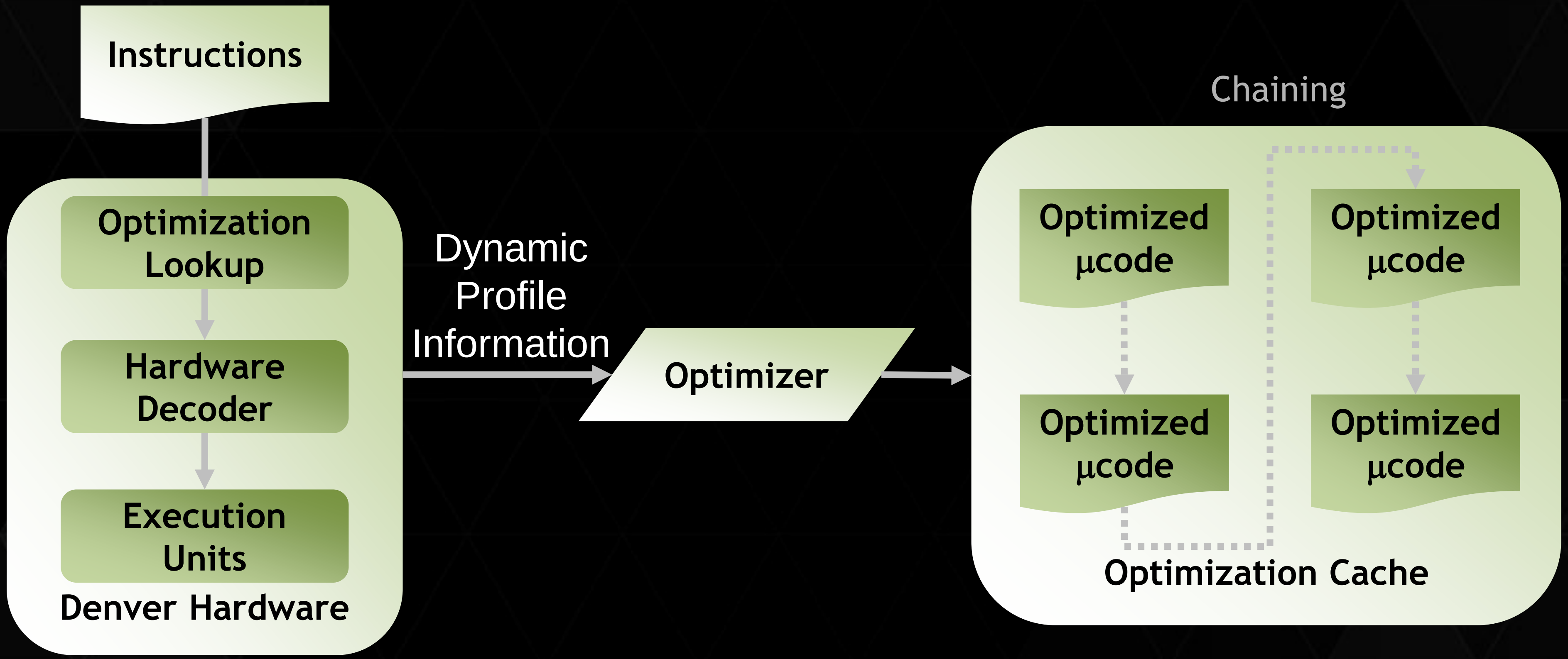
- Unrolls Loops
- Renames registers
- Reorders Loads and Stores
- Improves control flow
- Removes unused computation
- Hoists redundant computation
- Sinks uncommonly executed computation
- Improves scheduling

DYNAMIC CODE OPTIMIZATION

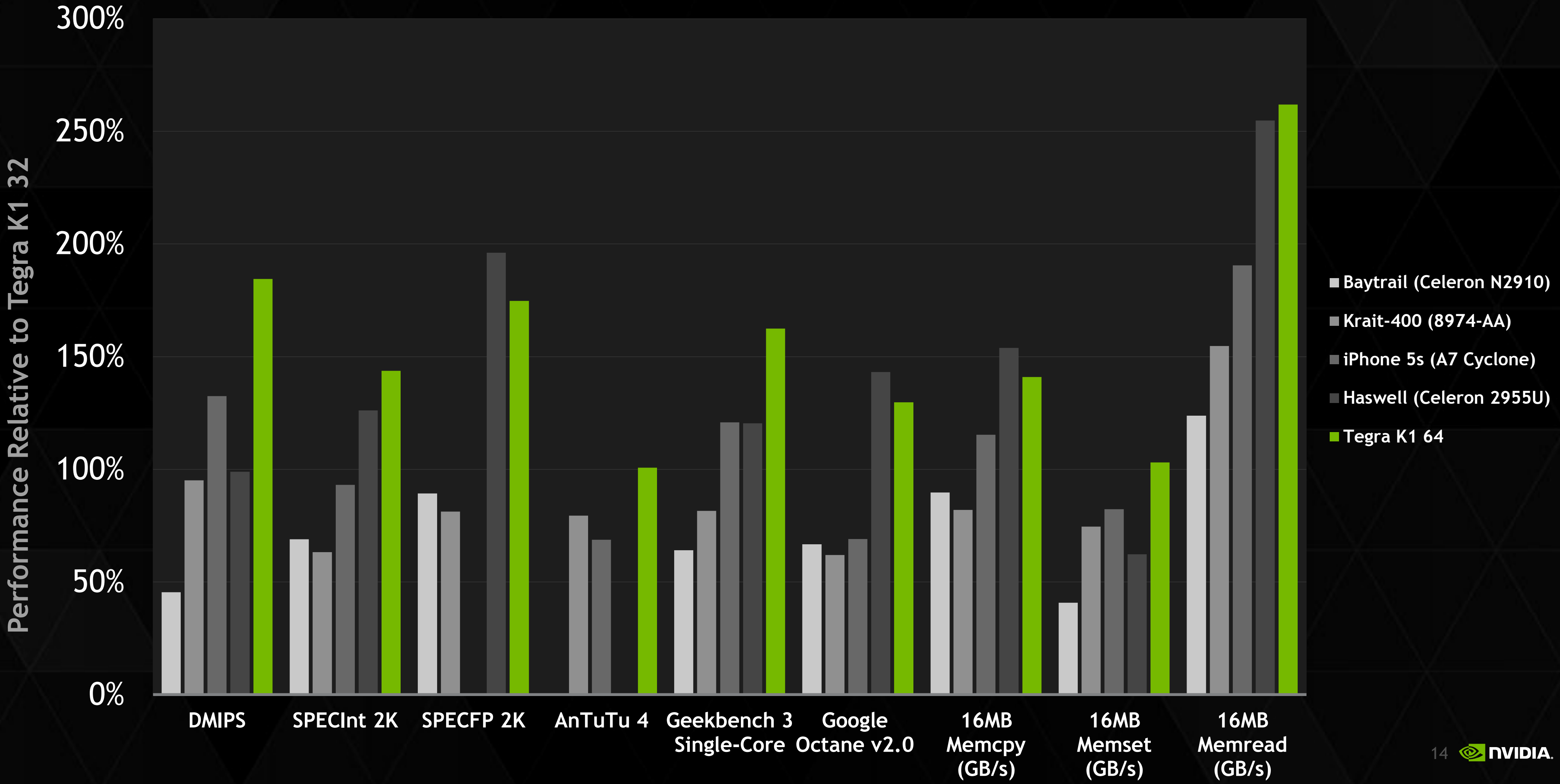
OPTIMIZE ONCE, USE MANY TIMES



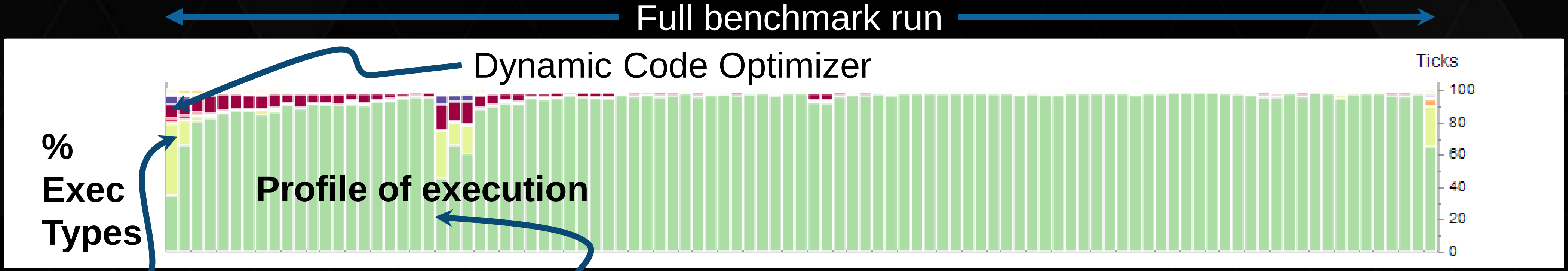
DYNAMIC CODE OPTIMIZATION
OPTIMIZE ONCE, USE MANY TIMES



DENVER PERFORMANCE



DCO: AN EXAMPLE SPECINT – CRAFTY EXECUTION

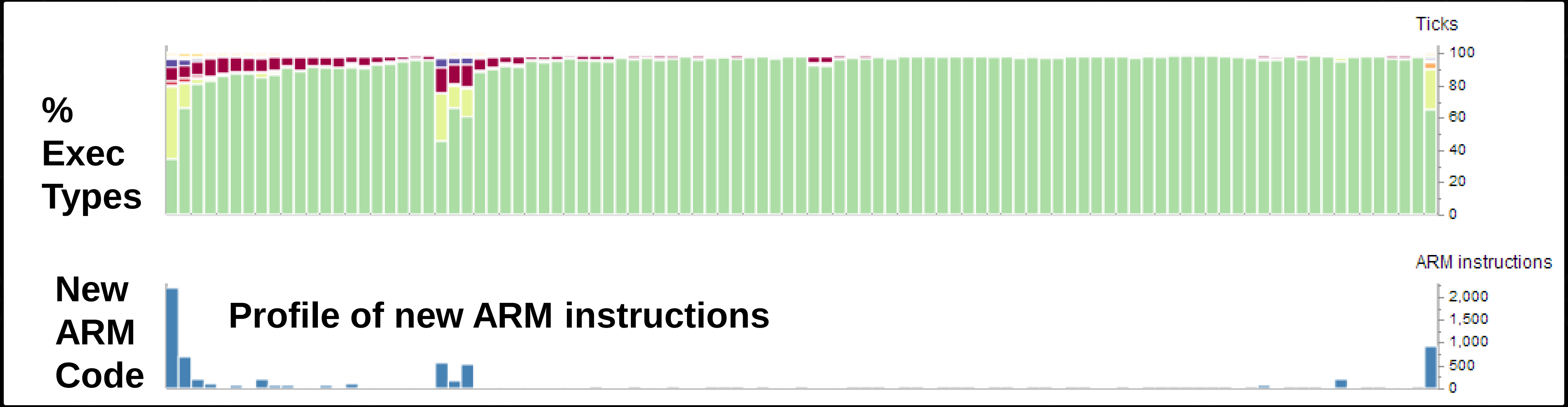


HW Decoder execution

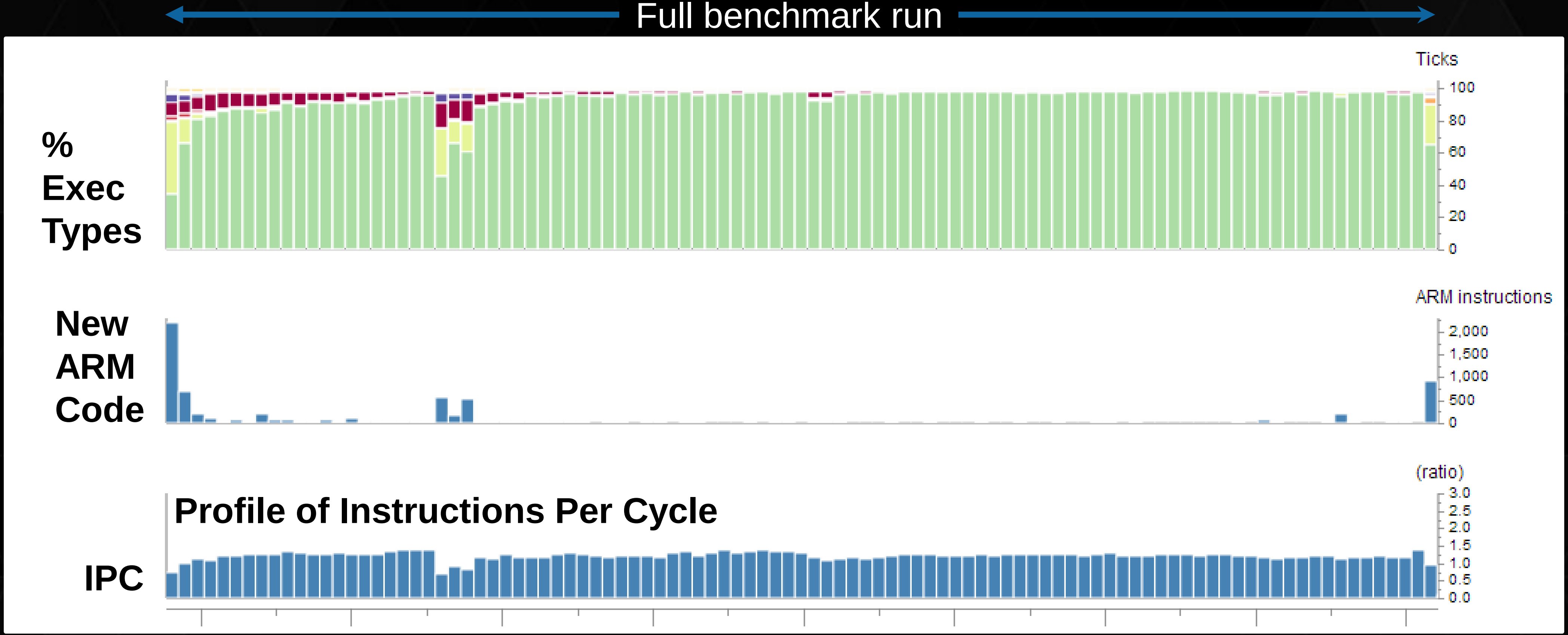
Optimized ucode execution

DCO: AN EXAMPLE SPECINT – CRAFTY EXECUTION

Full benchmark run

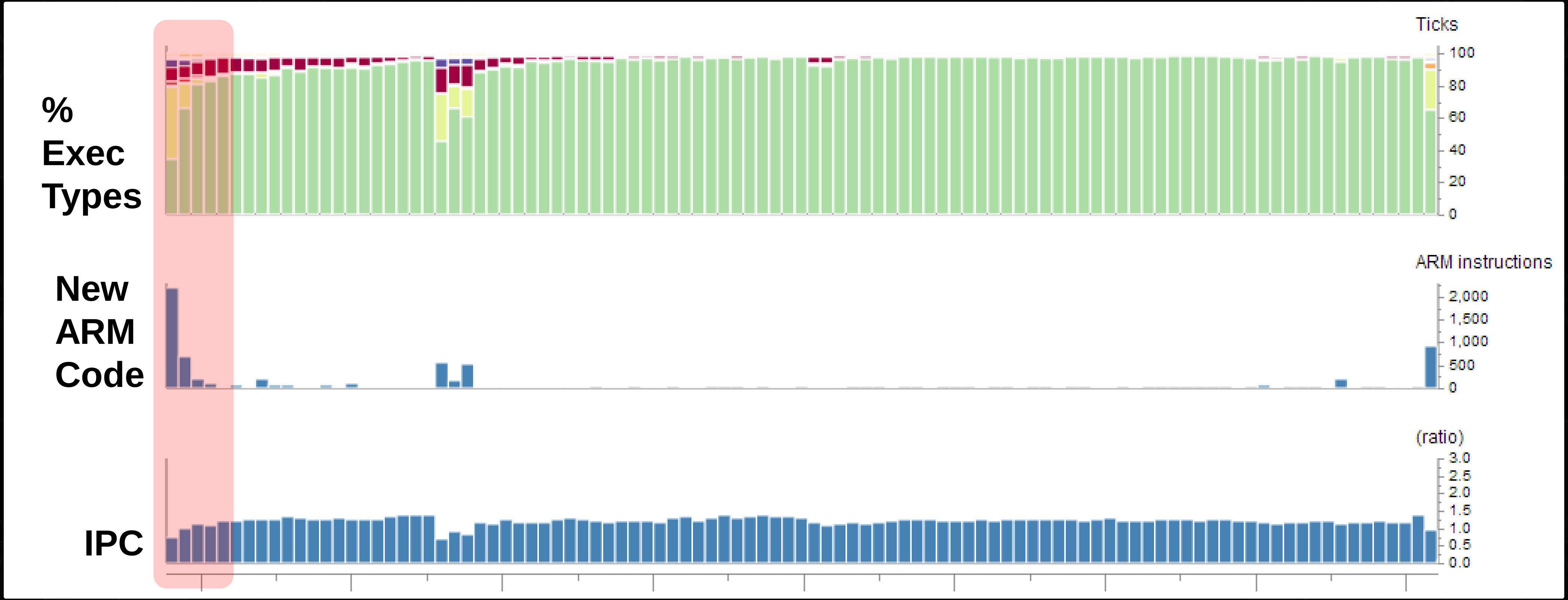


DCO: AN EXAMPLE SPECINT – CRAFTY EXECUTION

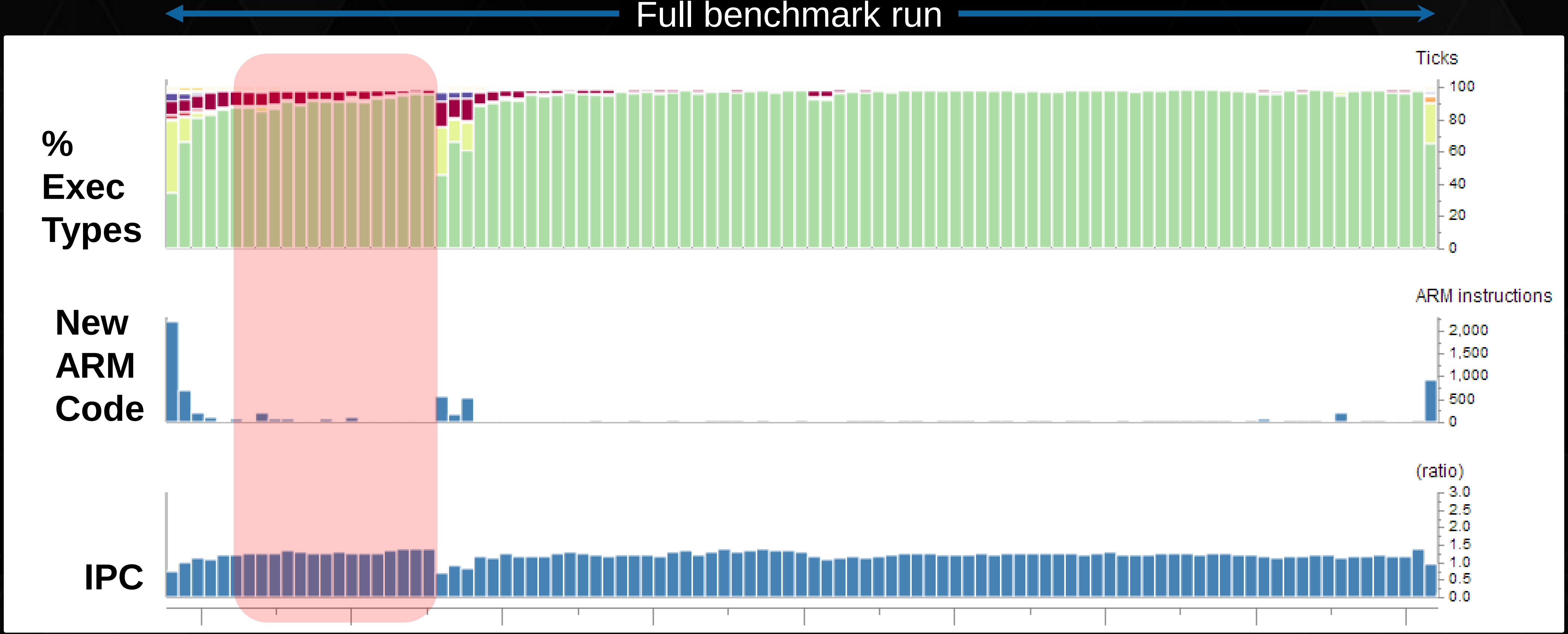


DCO: AN EXAMPLE SPECINT – CRAFTY EXECUTION

Full benchmark run

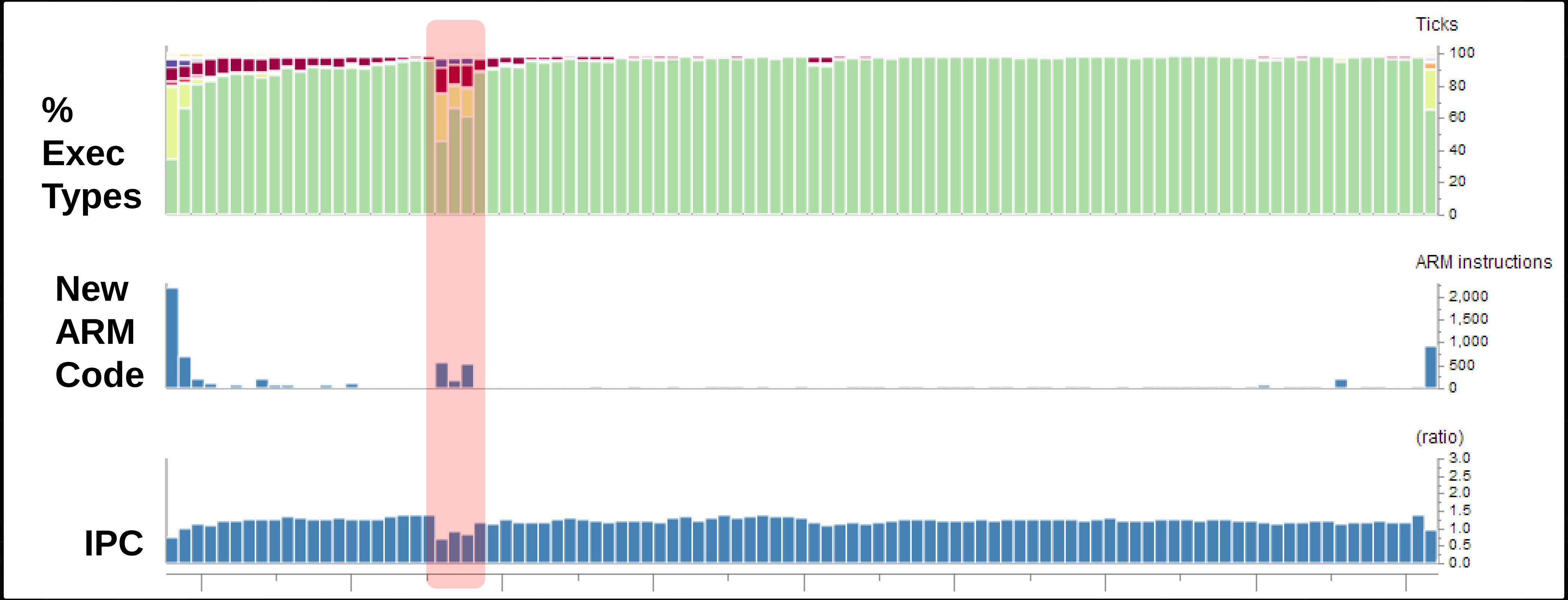


DCO: AN EXAMPLE SPECINT – CRAFTY EXECUTION

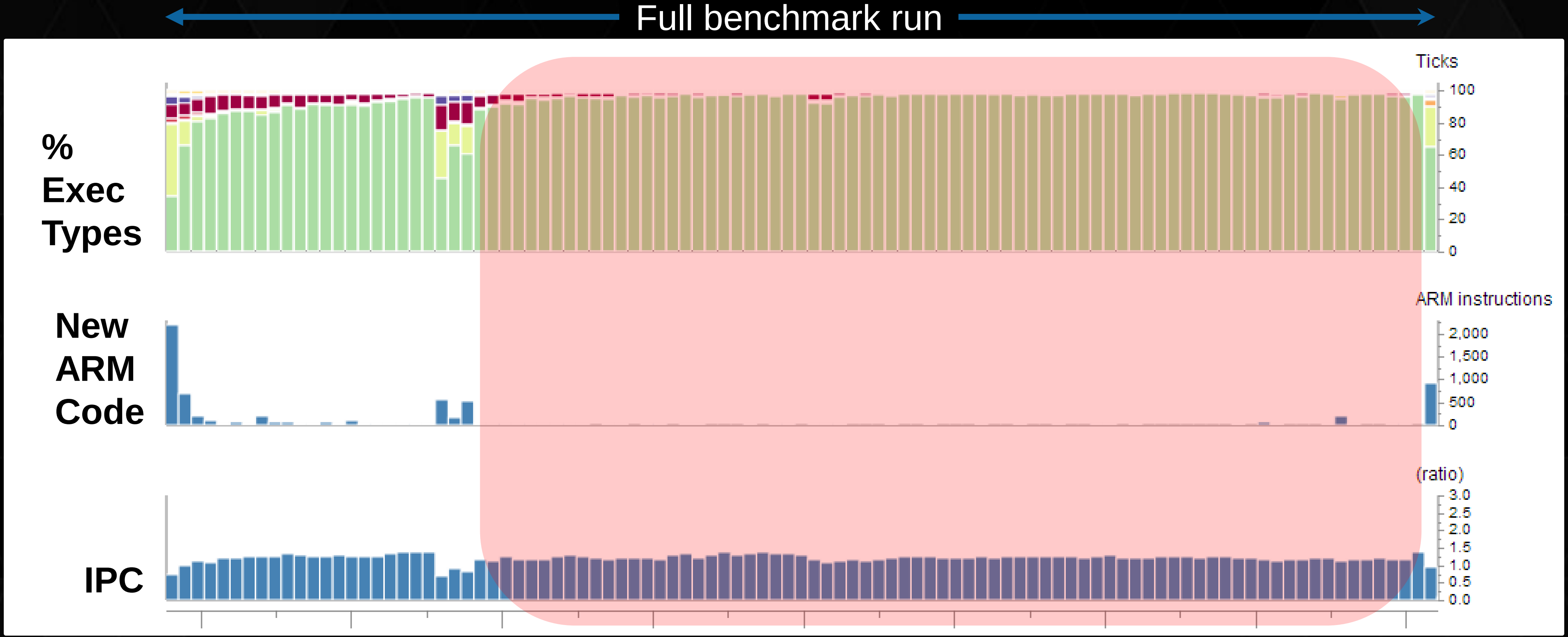


DCO: AN EXAMPLE SPECINT – CRAFTY EXECUTION

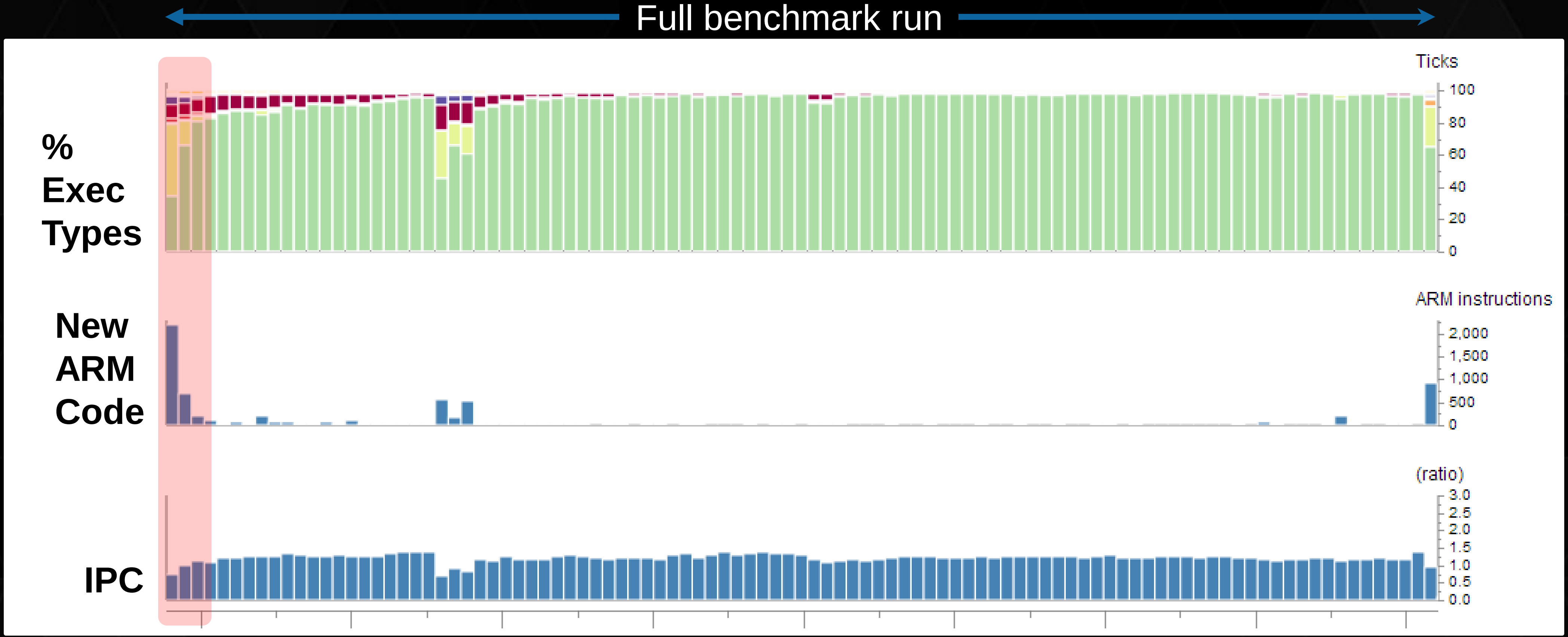
Full benchmark run



DCO: AN EXAMPLE SPECINT – CRAFTY EXECUTION

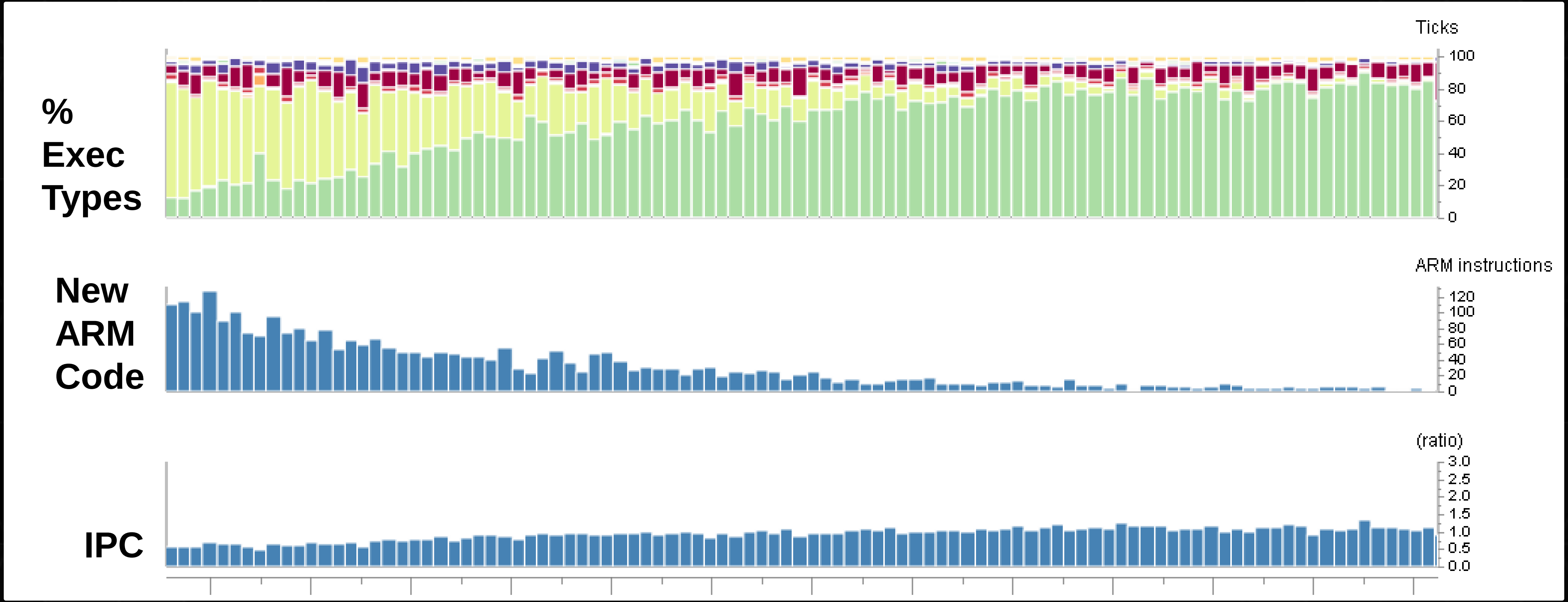


DCO: AN EXAMPLE SPECINT – CRAFTY EXECUTION



DCO: AN EXAMPLE SPECINT – CRAFTY EXECUTION

3% of benchmark run



CONCLUSION

- Dynamic Code Optimization is the architecture of the future
  - Breaks the out-of-order window physical limitation
  - Opens synergy between HW and SW that current architectures lack
  - Improves efficiency by optimizing once and using many times
- Delivering PC-class performance to mobile form factors
  - Enables PC-class gaming experience
  - Enables true enterprise applications
  - Enables content creation

ACKNOWLEDGMENT

- We would like to thank the CPU team in NVIDIA for all the creativity, hard work, and dedication to bring this vision to a reality.