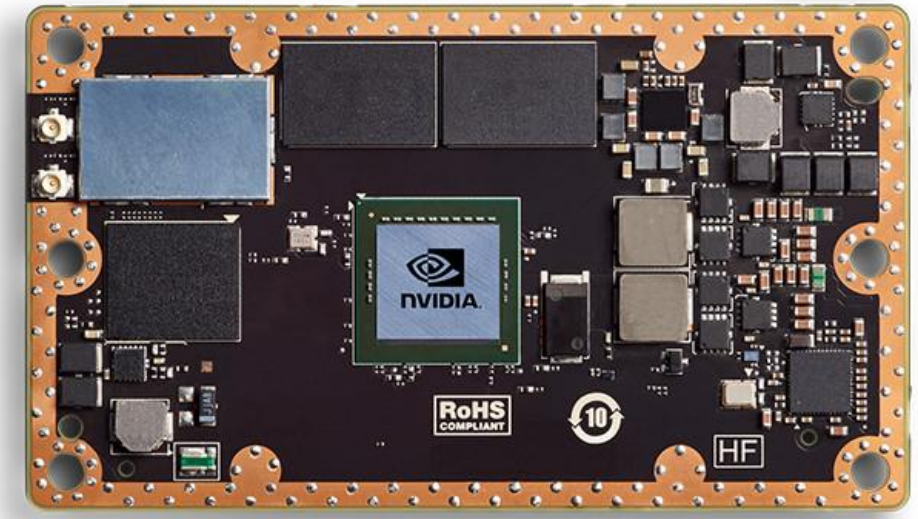
Hercules project

EU funded H2020
2016 – 2018
hercules2020.eu

Using COTS multicore for real-time embedded systems

Provide predictable timing guarantees when multiple cores may contend for shared resources, like memory and L2 cache.

Hercules project: architecture



Jetson TX1

GPU: NVIDIA Maxwell ™, 256 CUDA cores

Quad **ARM®** A57

16-way set associative 2MB L2 cache

Pseudo-random replacement policy

No lockdown mechanism

EVIDENCE®
EMBEDDING TECHNOLOGY

ERIKA 3

Erika Enterprise

First open-source RTOS certified for automotive (OSEK/VDX)

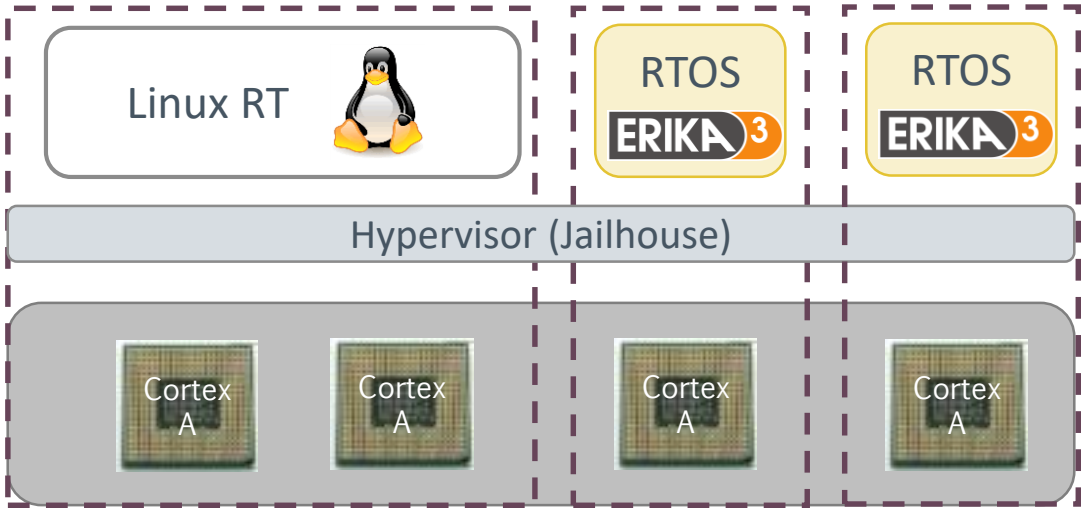
Implements the **AUTOSAR** OS SC1 API

Open-source GPL and commercial licenses

http://www.erika-enterprise.com

https://github.com/evidence/erika3

Hercules project: architecture

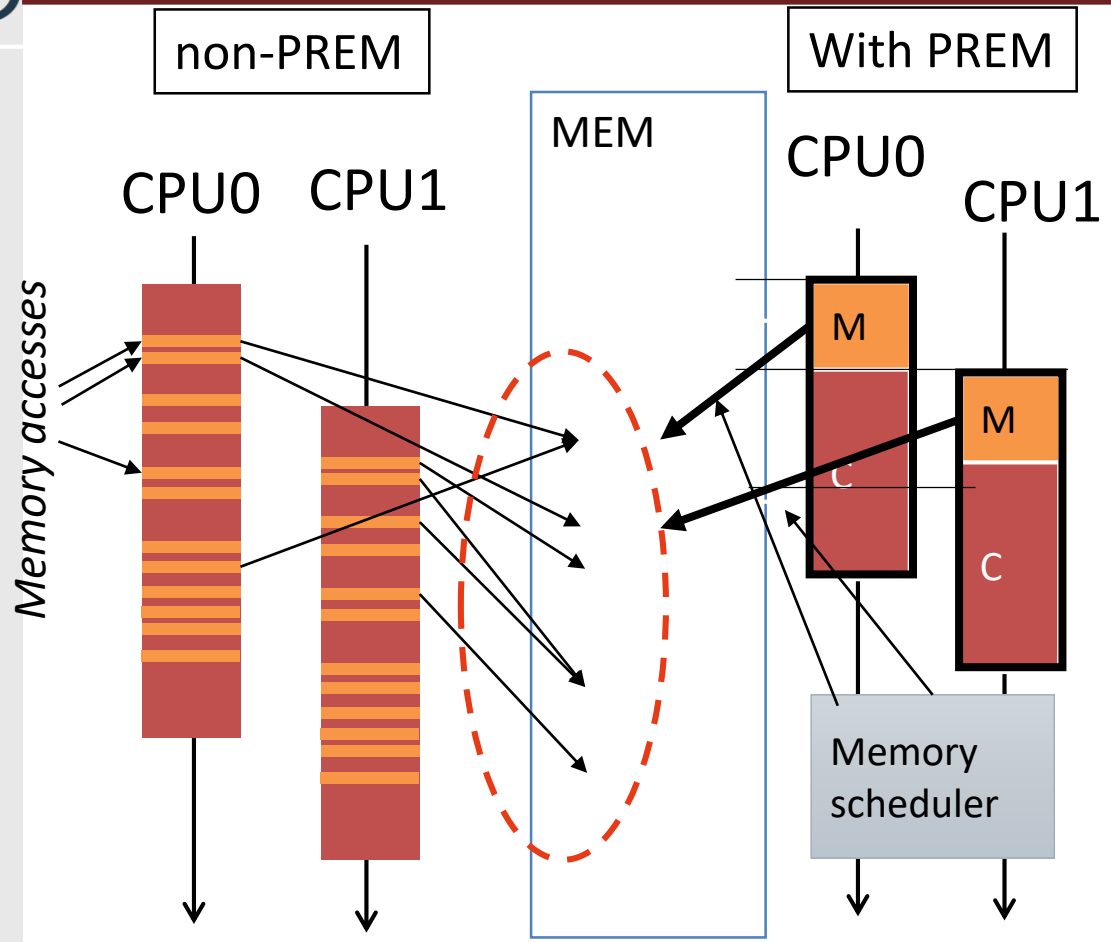


- Porting of the RTOS open-source **ERIKA** Enterprise on TX1/TX2/Ultrascale
- Porting the **Jailhouse** hypervisor on Nvidia Jetson TX1/TX2
- Implementing GRUB-PA in Linux kernel

Jailhouse partitioning Hypervisor based on Linux (https://github.com/siemens/jailhouse)

Our goal: Extend Jailhouse to schedule memory accesses and divide shared L2 cache

Hercules project: PREM model



Tasks divided into chunks of execution
Each chunk has two phases:

- A memory phase (M-phase) needed instruction and data are pre-fetched into local memory (e.g. cache)
- An execution phase (C-phase) tasks execute with **no conflicts**

PREM Compiler support needed

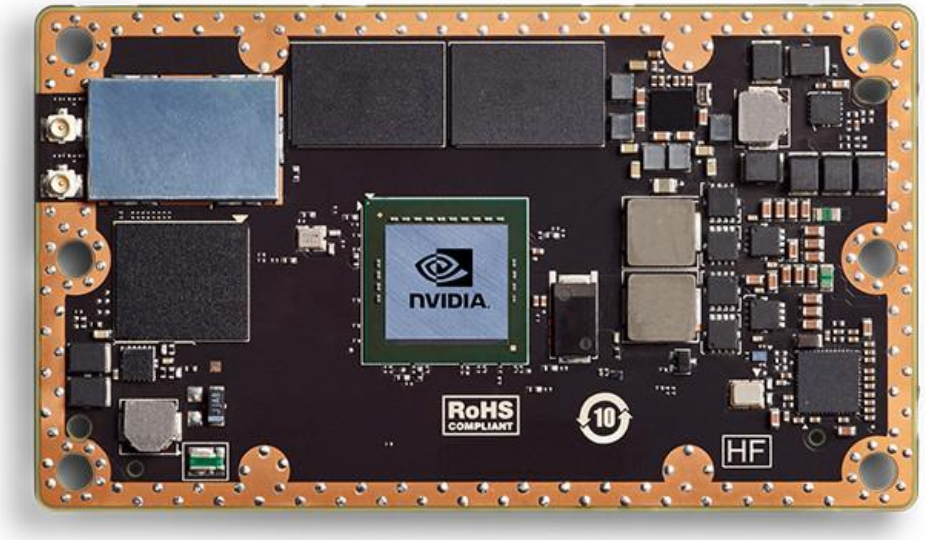
Rodolfo Pellizzoni, Emiliano Betti, Stanley Bak, Gang Yao, John Criswell, Marco Caccamo, Russell Kegley: A Predictable Execution Model for COTS-Based Embedded Systems.

Björn Forsberg, Andrea Marongiu, Luca Benini: GPUguard: Towards supporting a predictable execution model for heterogeneous SoC

Goals

PREM: During memory phase data must be prefetched into L2 cache

I How to divide set-associative cache among different cores => cache coloring
II How to use pseudo-random replacement policy in deterministic way => preventive invalidation



Jetson TX1

GPU: NVIDIA Maxwell ™, 256 CUDA cores

Quad **ARM®** A57

16-way set associative 2MB L2 cache

Pseudo-random replacement policy

No lockdown mechanism

Cache coloring in Jailhouse

Cache coloring

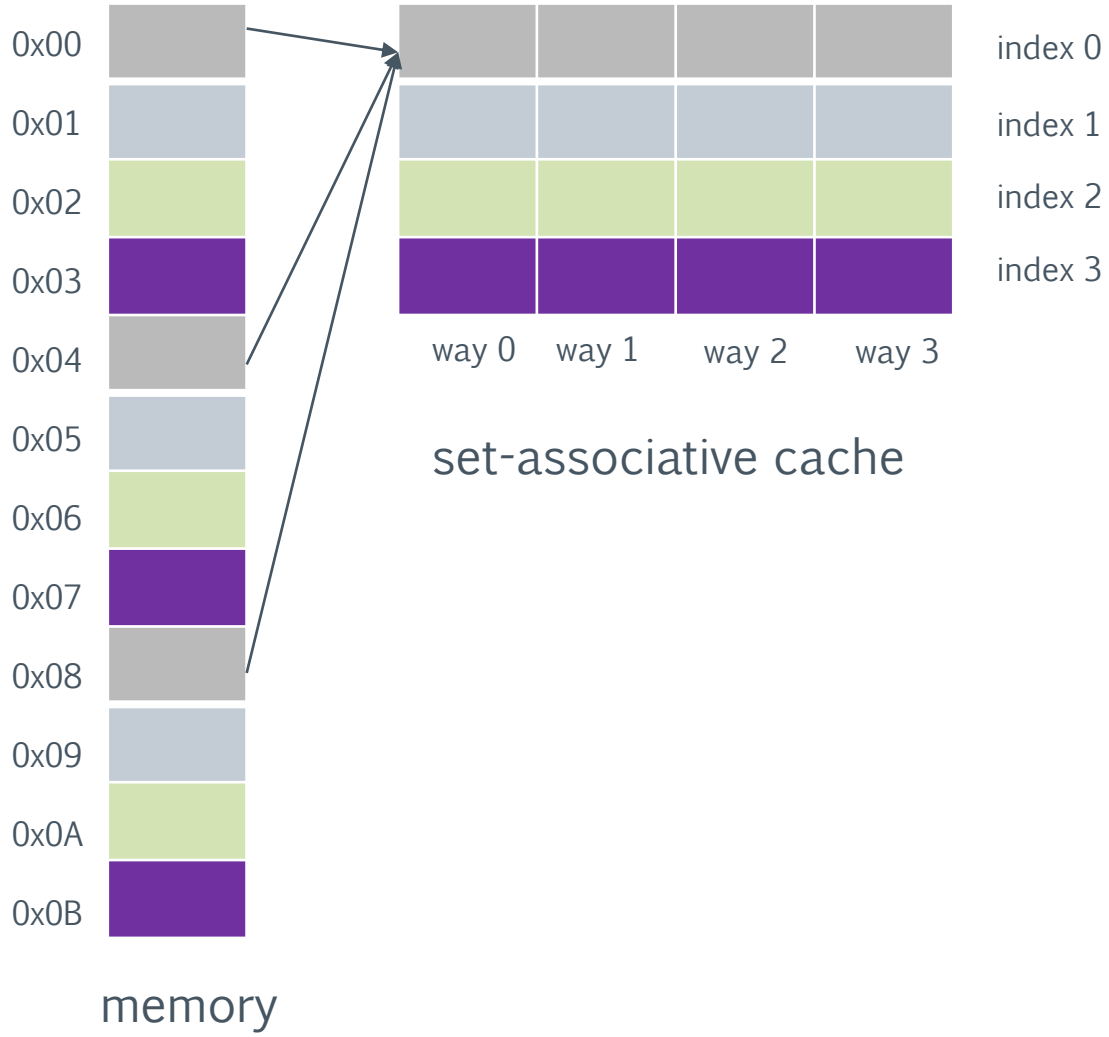
The cache is physically indexed.

The cache set index is assigned based on the memory address.

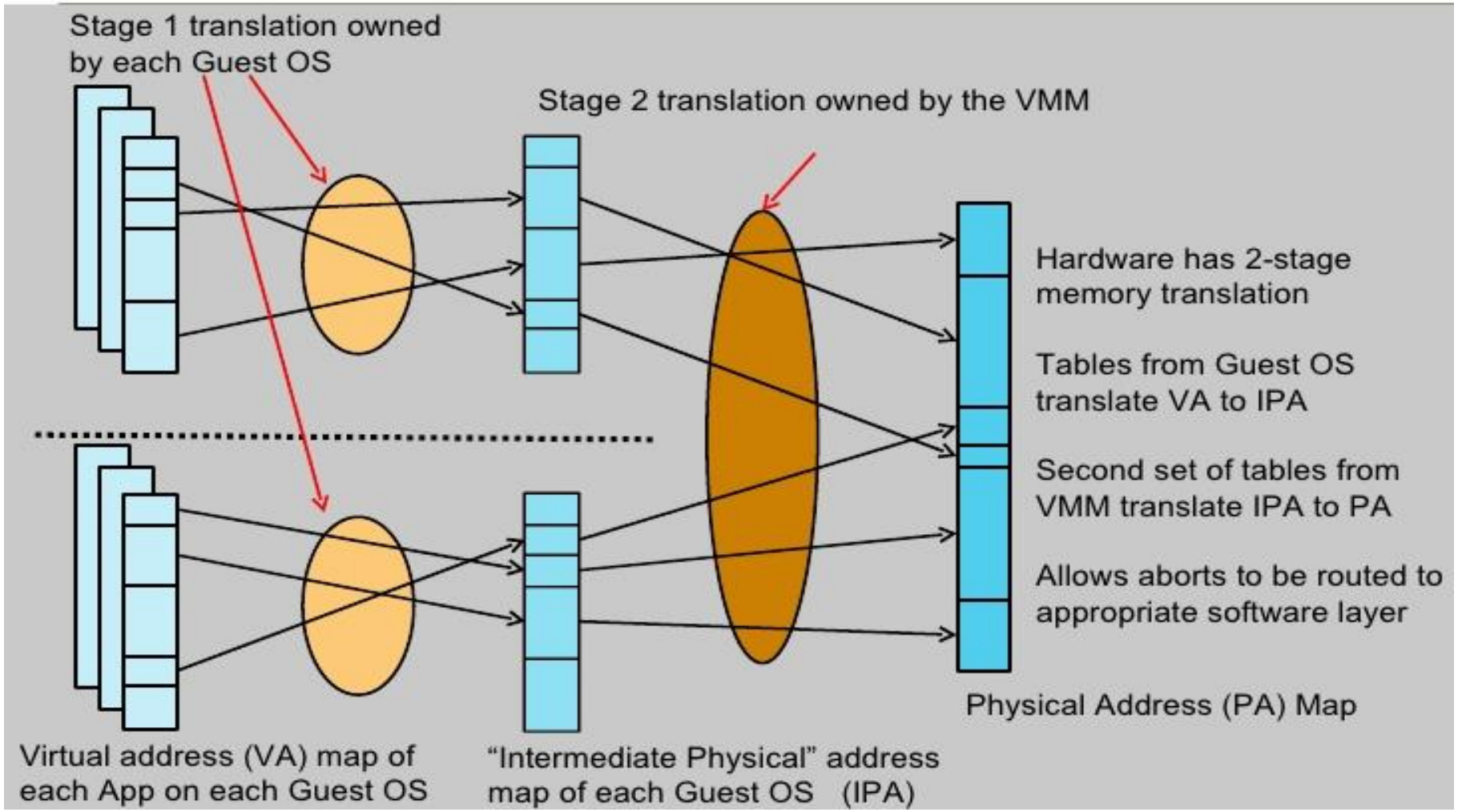
Physical memory pages are "colored" so that pages with different "colors" have different positions in CPU cache memory.

ARM A-57
2MB L2 cache 16 ways

Index = Address % 128KB
2MB/16=128KB



ARM Virtualization Extension



John Goodacre: Hardware Accelerated Virtualization in the ARM Cortex Processors

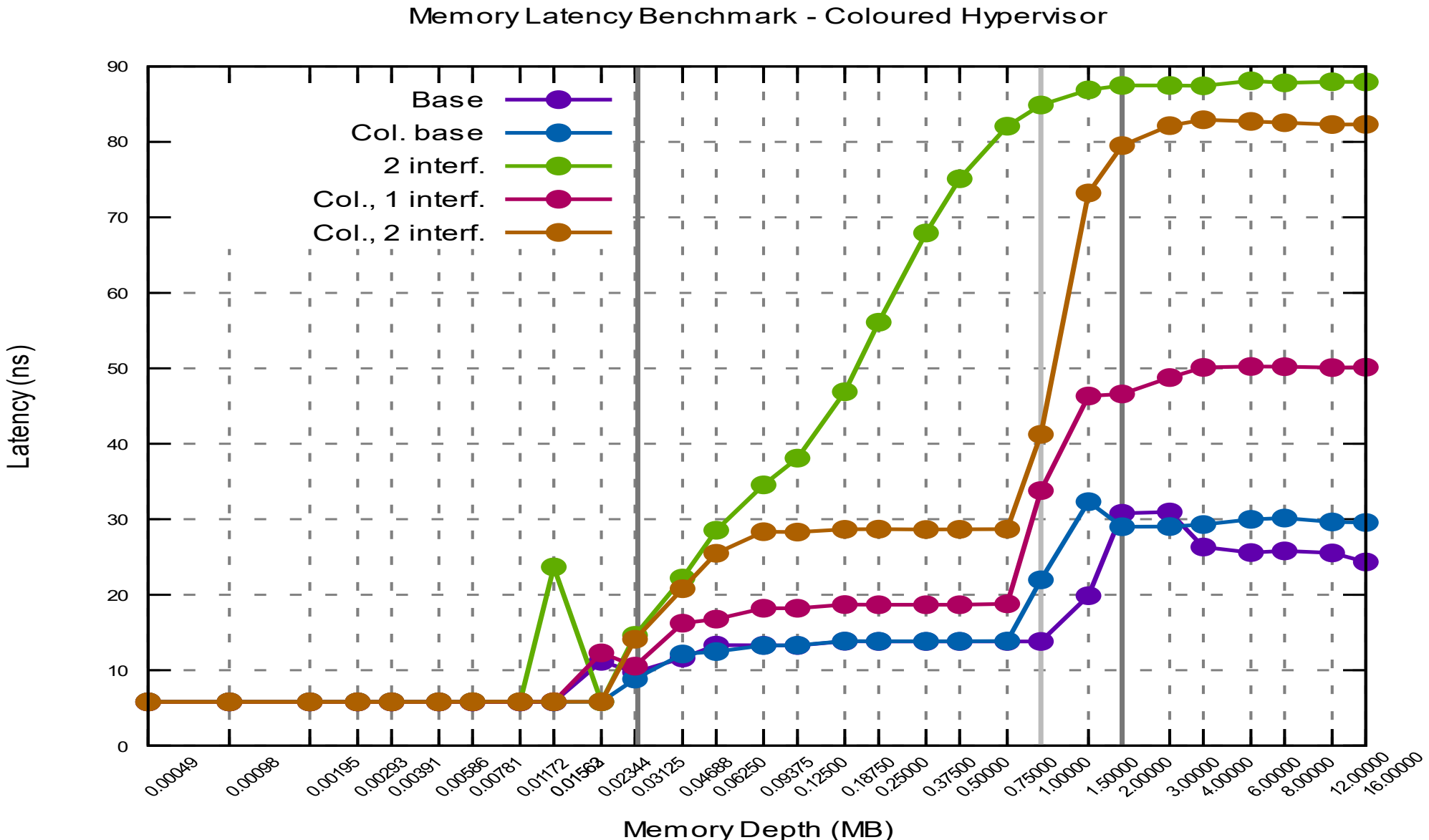
Cache coloring in Jailhouse

- memory region structure for memory locations of a given color
- image loading mechanism for non continuous memory regions
- automatic color generation as a function of cache parameters.

```
.memory_regions_colored = {
  {
    /* RAM */
    .memory = {
      .phys_start =   0x800620000,
      .virt_start =      0x20000,
      .size       =       0x20000,
      .flags      = JAILHOUSE_MEM_READ |
                    JAILHOUSE_MEM_WRITE |
                    JAILHOUSE_MEM_EXECUTE |
                    JAILHOUSE_MEM_LOADABLE,
    },
    .colors = 0x00ff, // color selection bit mask
  },
},
```

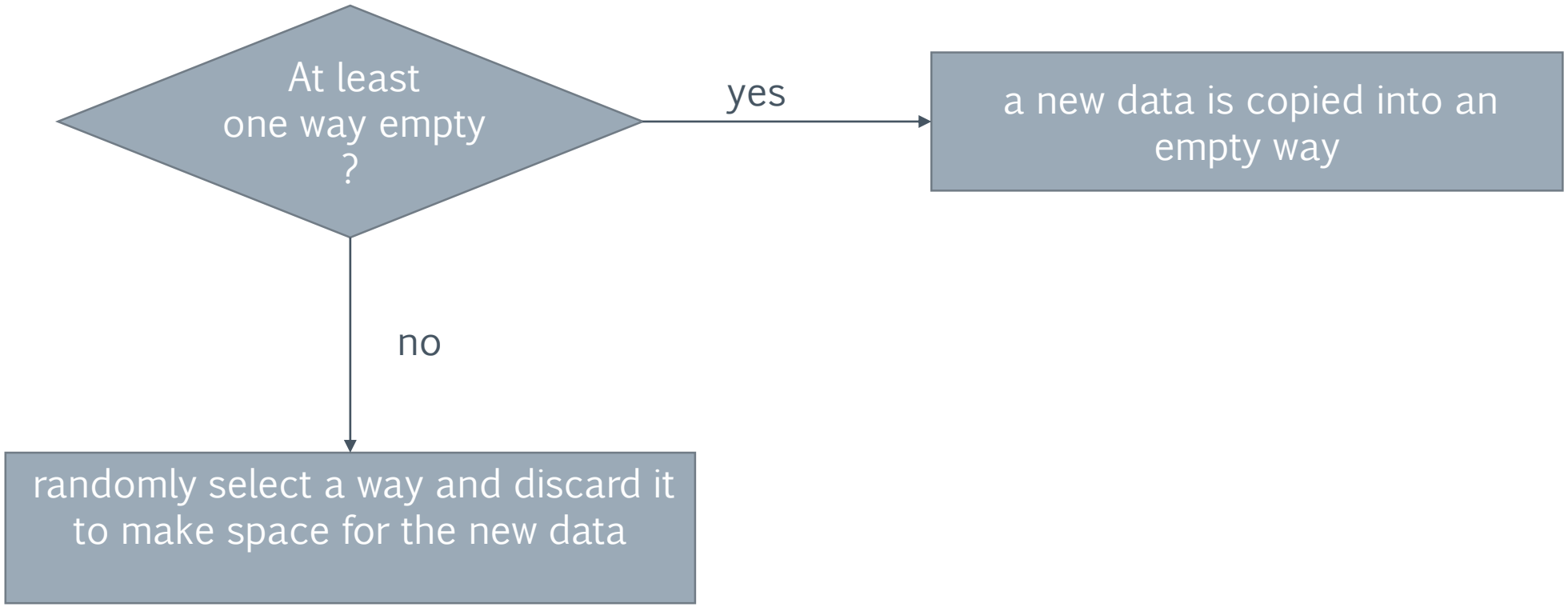
Paolo Modica, Alessandro Biondi, Giorgio Buttazzo, Anup Patel: Supporting Temporal and Spatial Isolation in a Hypervisor for ARM Multicore Platforms

Cache coloring in Jailhouse: LMbench

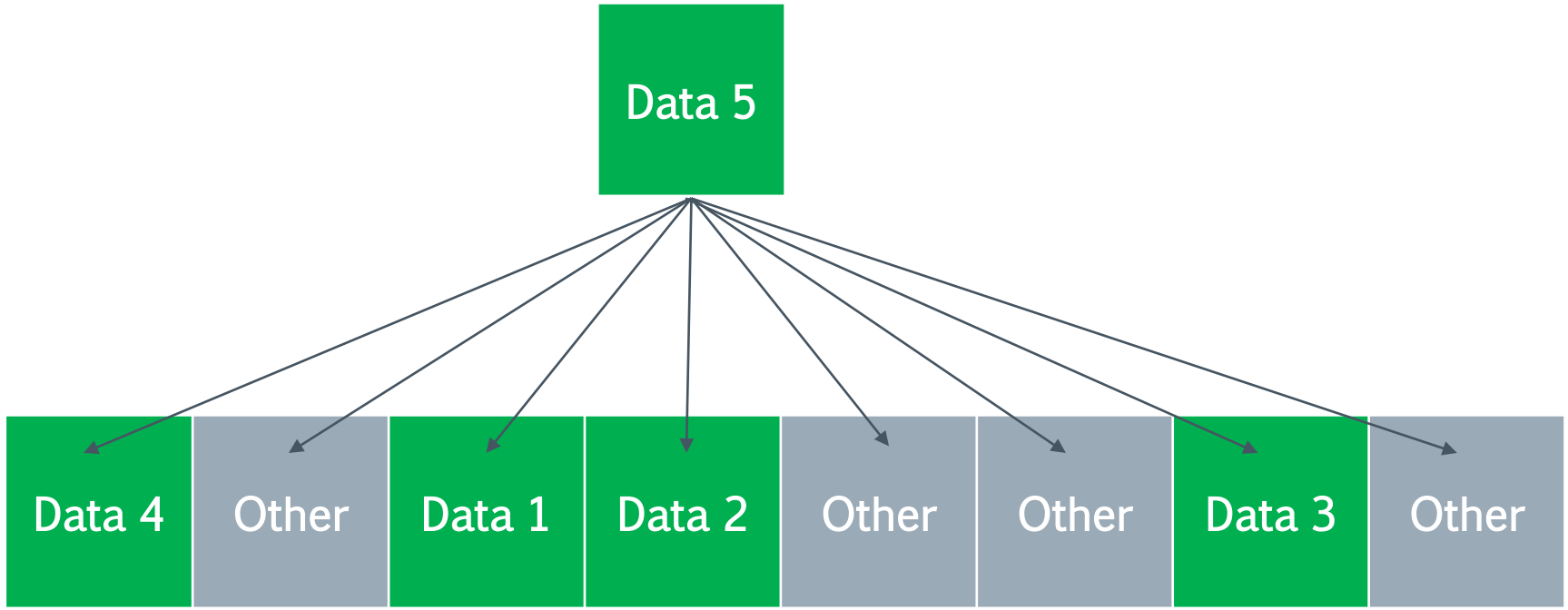


Preventive Invalidation

Pseudo-random cache



Pseudo-random cache: self-eviction



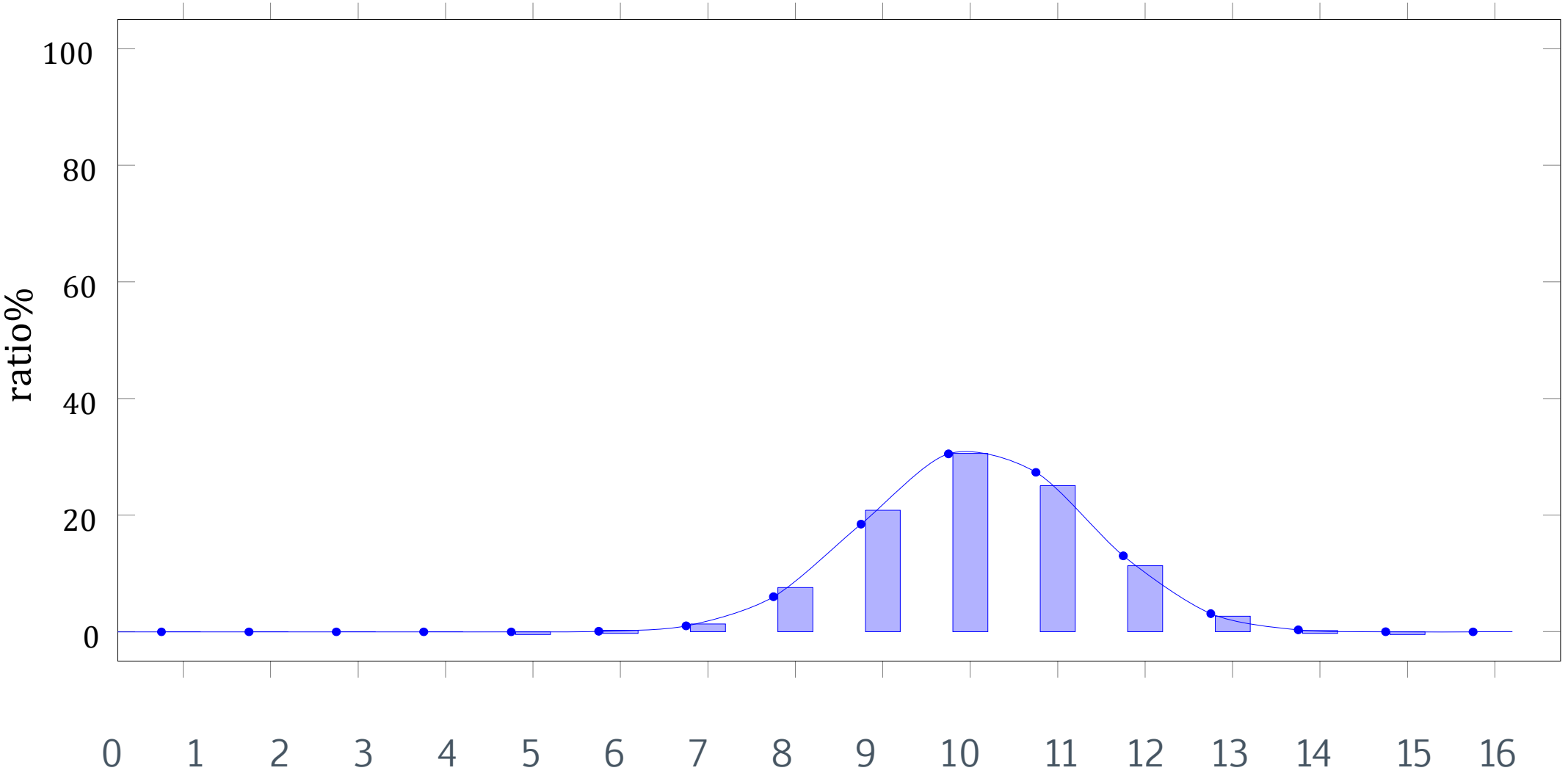
8-way set associative cache
4 Data in use (green), 4 stall data (grey)
Prefetching new data: 50% chances to evict data that is in use

Pseudo-random cache: measure self-eviction

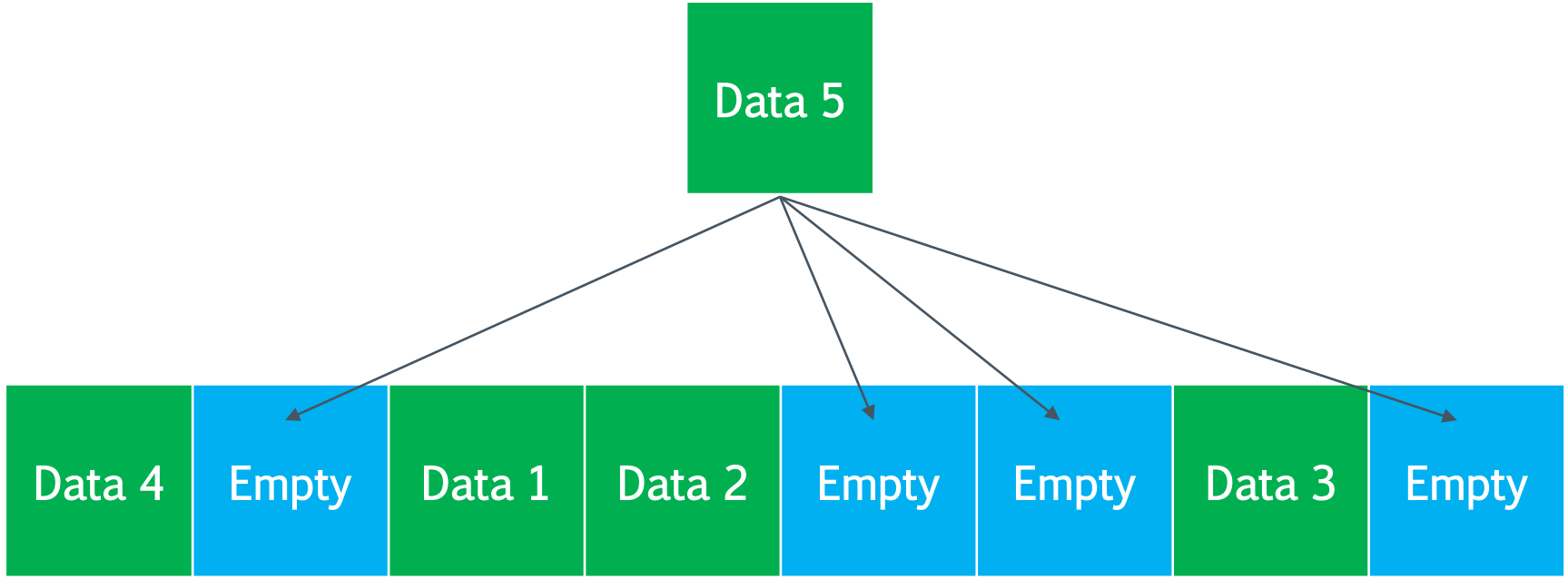
map Data_1, Data_2, ..., Data_16 to the different memory regions with the same index

```
PRFM  PLDL2KEEP, Data_1;
PRFM  PLDL2KEEP, Data_2;
PRFM  PLDL2KEEP, Data_3;
...
PRFM  PLDL2KEEP, Data_16;

DSB  ISH;

for(way=0; way<16; way++) {
        RAMINDEX = (0x10 <<  24) + (way <<  18) + (index <<  6);
        SYS  #0, c15, c4, #0, RAMINDEX;
}
```

Pseud-random cache: measure self-eviction



Pseudo-random cache: preventive invalidation



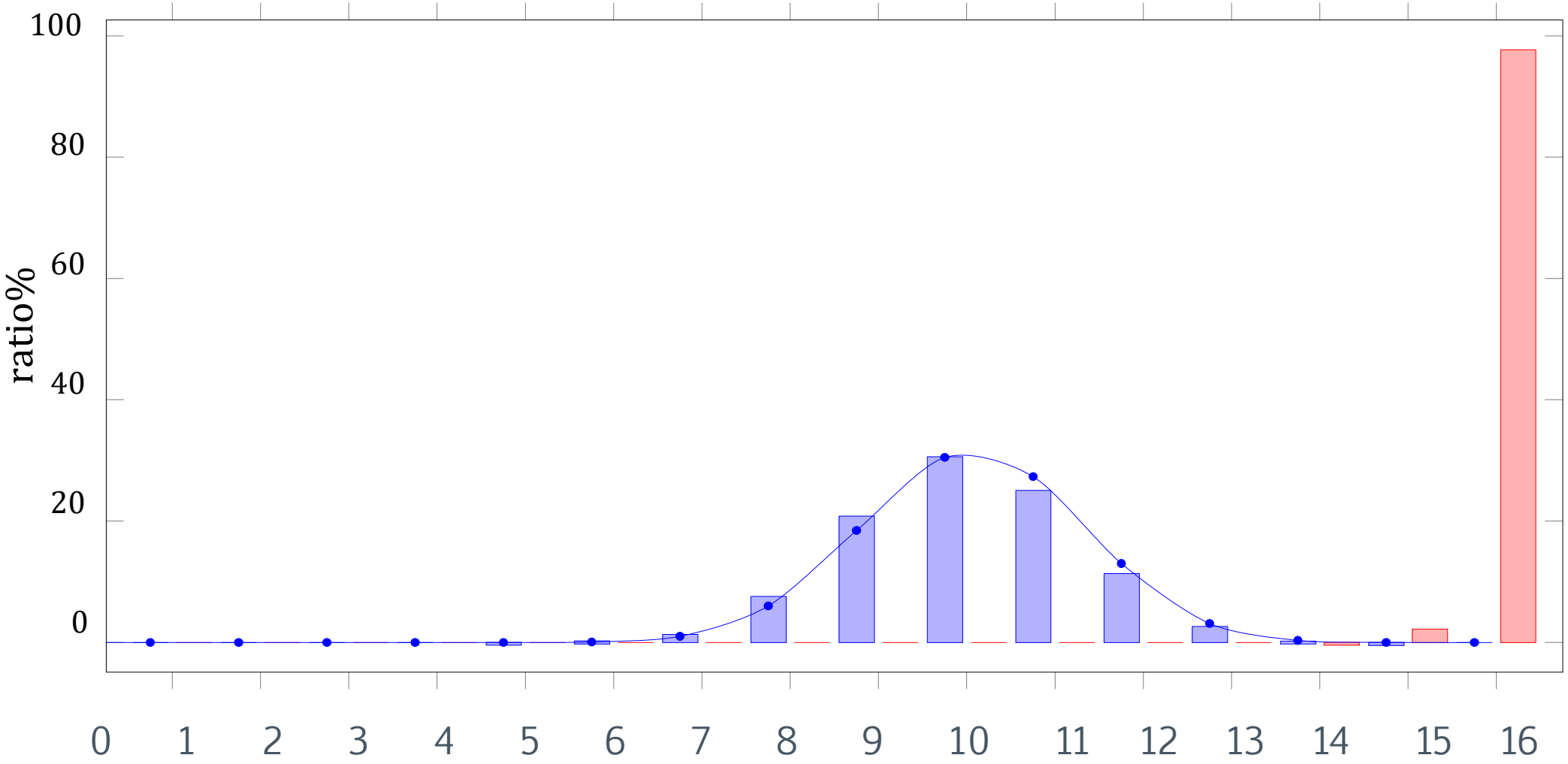
8-way set associative cache
4 Data in use (green), 4 empty ways (blue)
Prefetching new data: 100% chances to avoid eviction of data in use

Pseudo-random cache: preventive invalidation

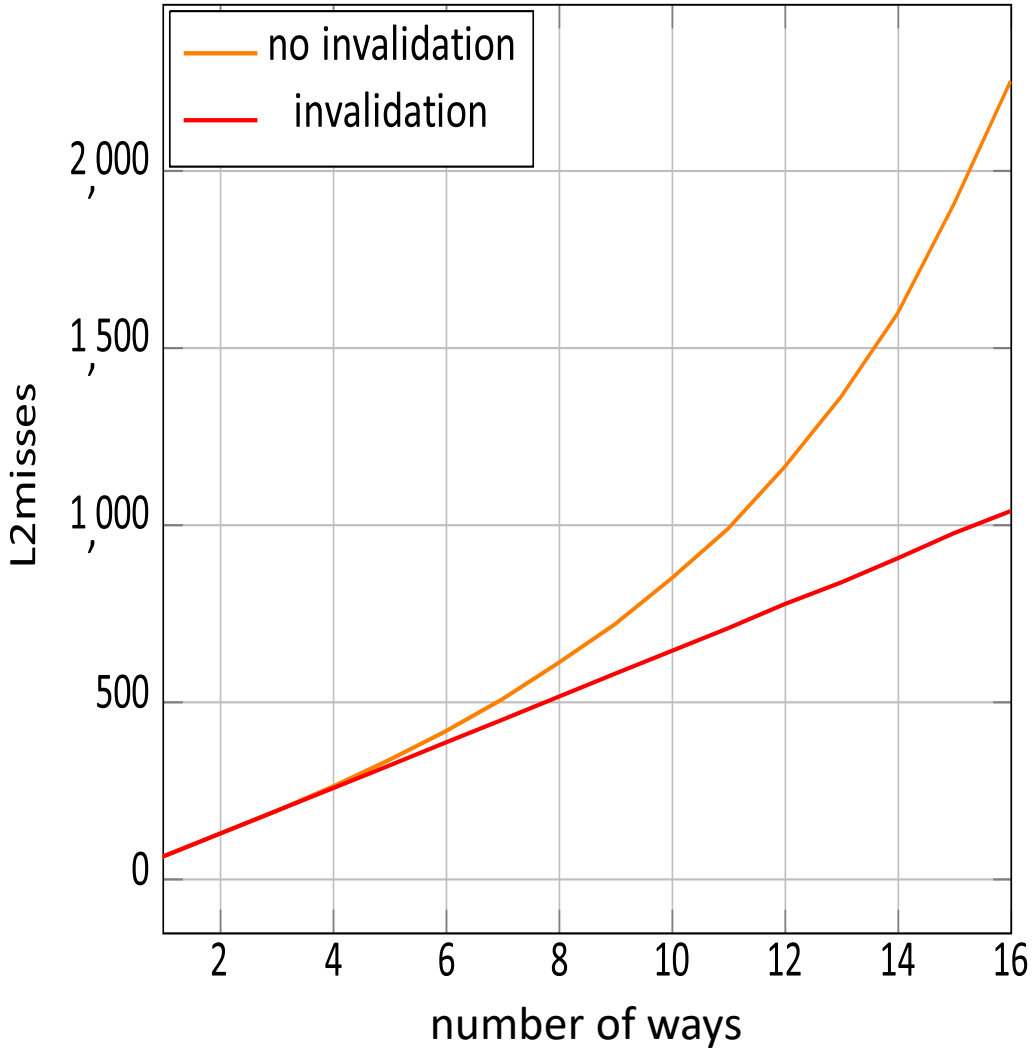
map Data_1, Data_2, …, Data_16 to the different memory regions with the same index

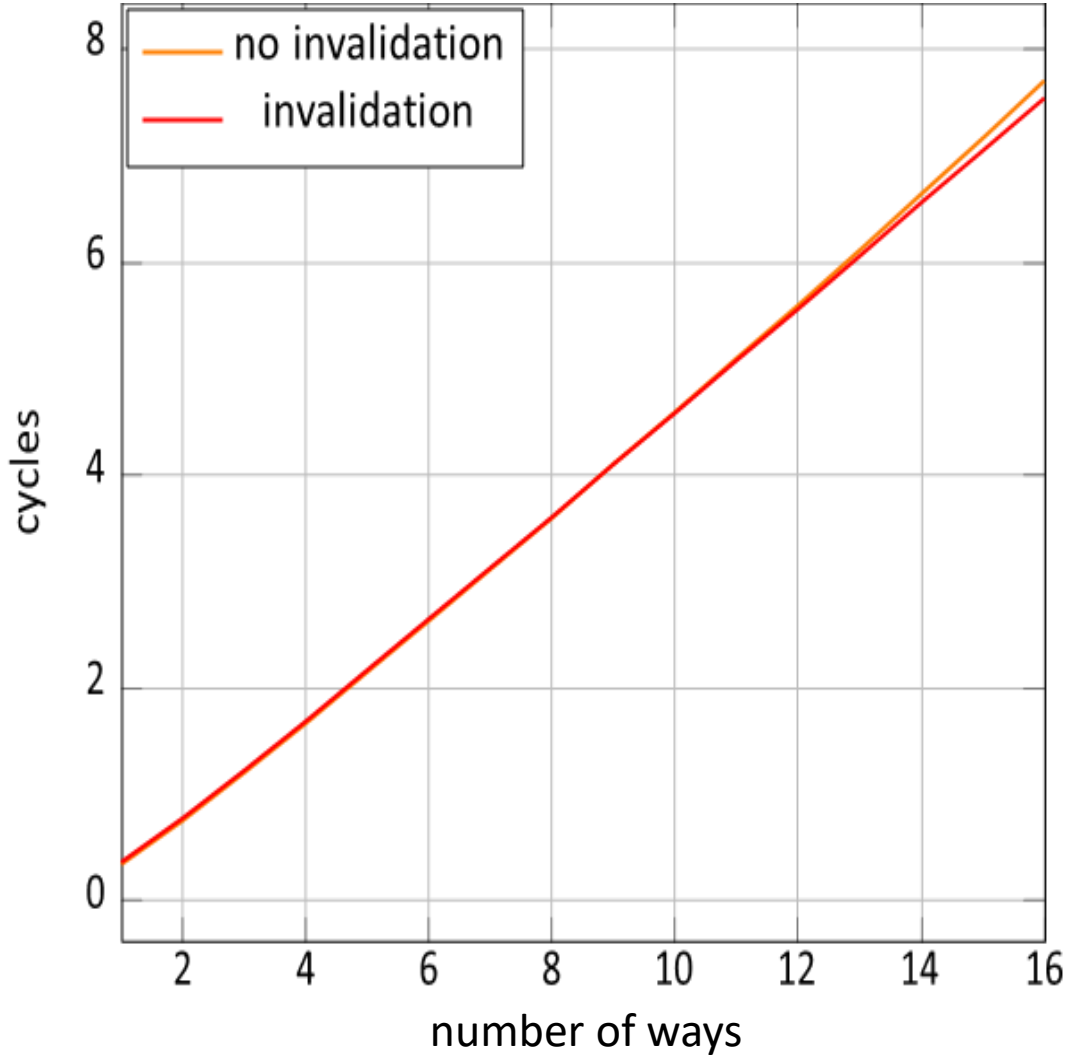
```
for(way=0; way<16; way++) {
        DC CISW, ((way<<28) + (index<<6) + 0x02);
}

PRFM  PLDL2KEEP, Data_1;
PRFM  PLDL2KEEP, Data_2;
PRFM  PLDL2KEEP, Data_3;
…
PRFM  PLDL2KEEP, Data_16;

DSB  ISH;

for(way=0; way<16; way++) {
        RAMINDEX = (0x10 <<  24) + (way <<  18) + (index <<  6);
        SYS  #0, c15, c4, #0, RAMINDEX;
}
```

Pseudo-random cache: preventive invalidation

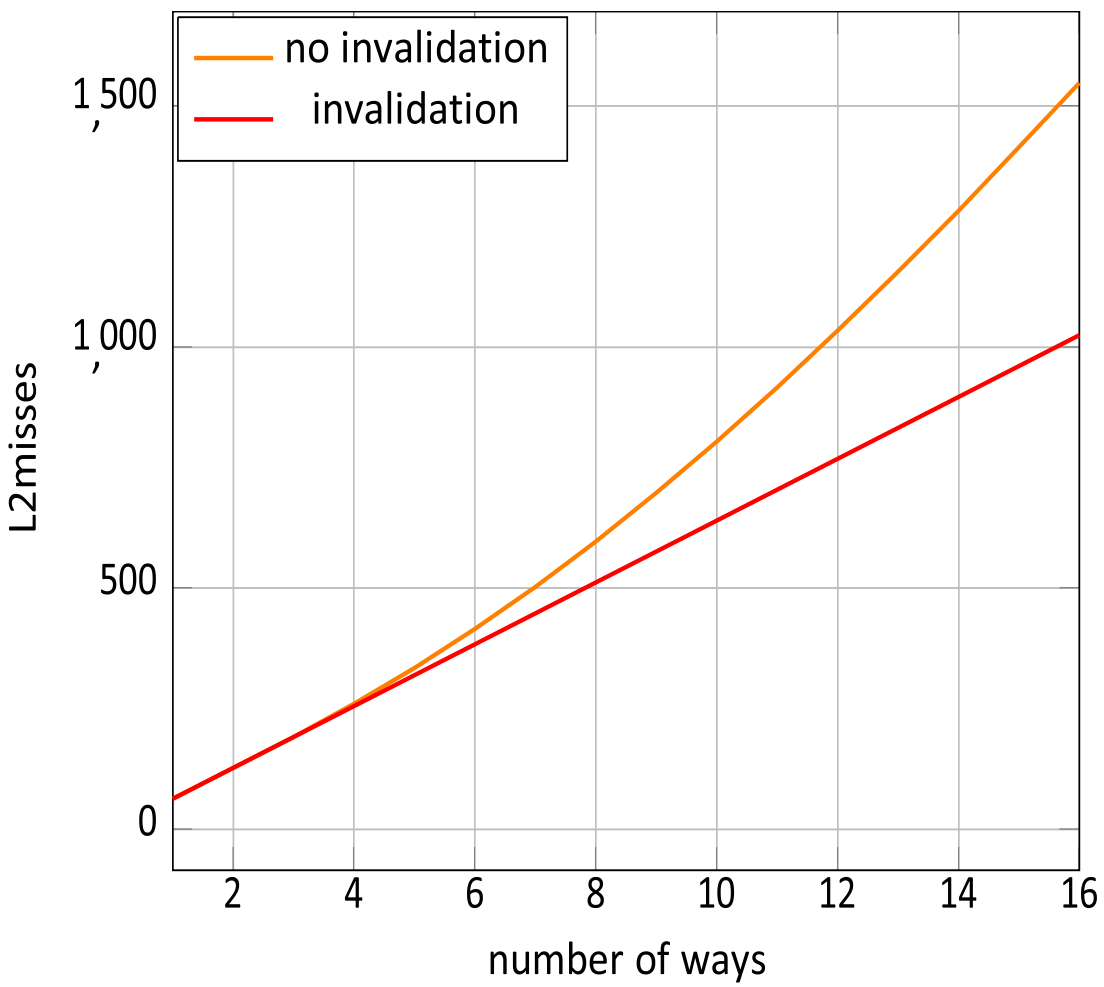


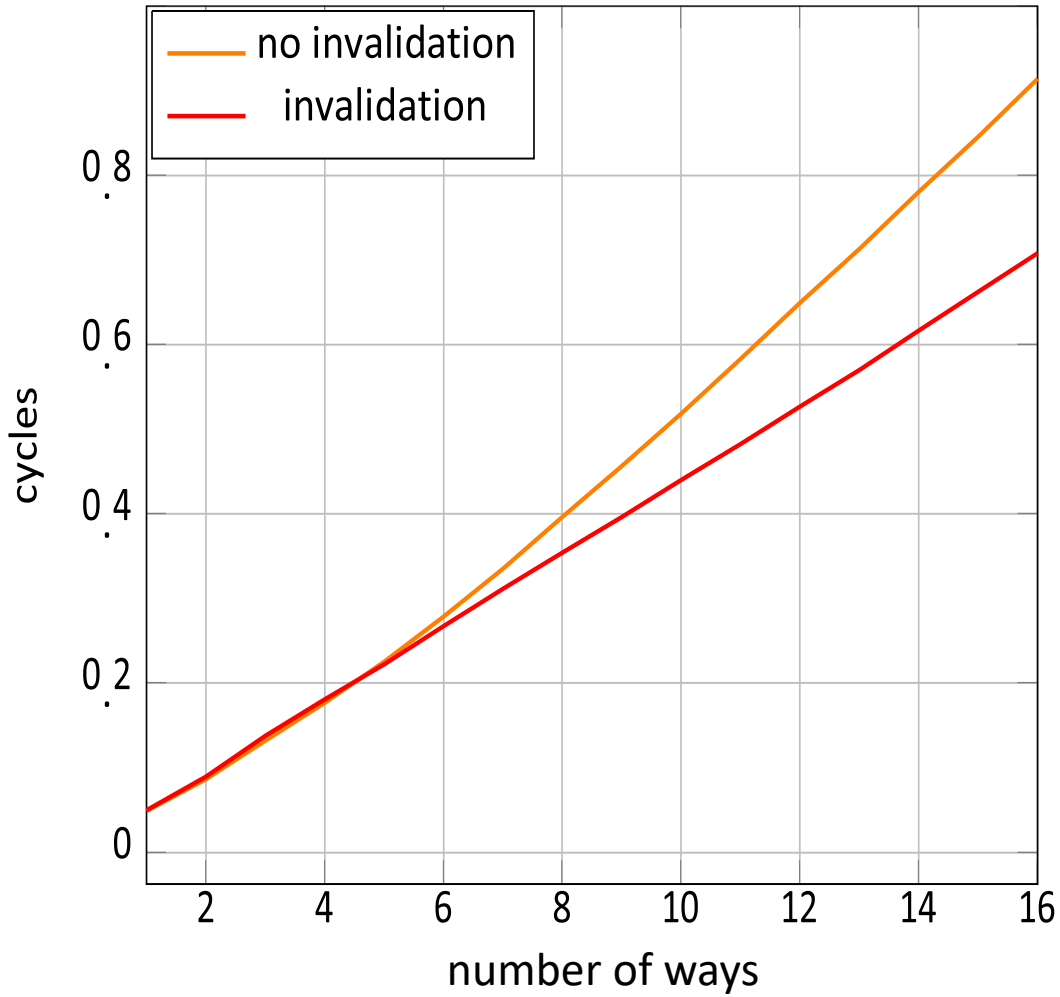
Benchmark – Heapsort (from Linux kernel)





Benchmark: two iterations of linked list





Pseudo-random cache: worst-case behavior

- Clean and Invalidate: every next access hit
- No Clean and Invalidate: every next access miss
  probability of 15 self-evictions in row (16 prefetches, each next evicting previous data) is very low: $(1/16)^{15}$
- Static Probabilistic Timing Analysis
  probability of the worst-case behaviors can be upper bounded at extremely low levels which are below the maximum permissible failure rate (e.g. 10^{-9} per hour)

Benjamin Lesage, David Griffin, Sebastian Altmeyer, Liliana Cucu-Grosjean, Robert I. Davis: On the analysis of random replacement caches using static probabilistic timing methods for multi-path programs.

Sebastian Altmeyer, Robert I. Davis: On the correctness, optimality and precision of Static Probabilistic Timing Analysis.

Sebastian Altmeyer, Liliana Cucu-Grosjean, Robert I. Davis: Static probabilistic timing analysis for real-time systems using random replacement caches.

Conclusions

hercules2020.eu

1. Erika Enterprise ported on TX1/TX2/Ultrascale
2. Jailhouse ported on TX1/TX2
3. PREM compiler support
4. Cache coloring in Jailhouse
5. Preventive invalidation for pseudo-random caches
6. Automotive and avionic use cases

EVIDENCE® EMBEDDING TECHNOLOGY

UNIMORE UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

ETH Zürich

CZECH TECHNICAL UNIVERSITY IN PRAGUE

AIRBUS GROUP

MAGNETI MARELLI

pitom think over movement

Conclusions

hercules2020.eu

Thank you