CISTER - Research Center in
Real-Time & Embedded Computing Systems

isep Instituto Superior de Engenharia do Porto

INESCTEC

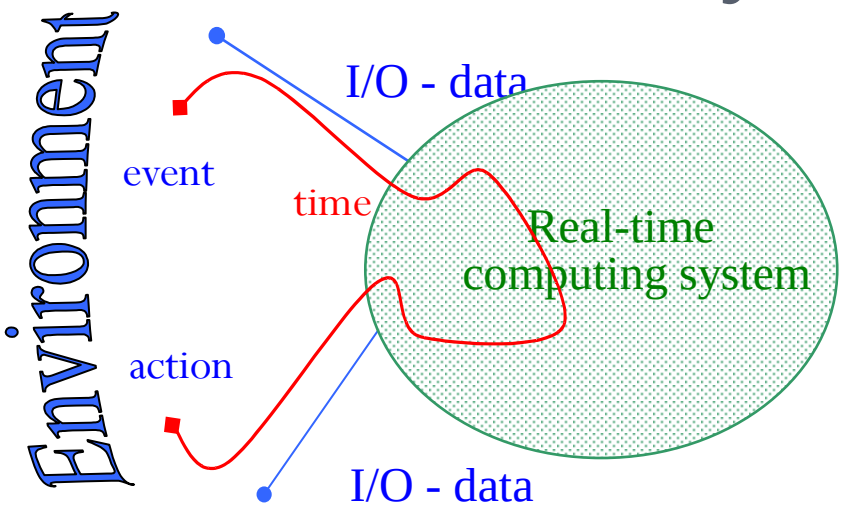
Identifying Sources of Unpredictability in COTS-based Multicore systems

Dakshina Dasari, Benny Akesson, Vincent Nelis,
Muhammad Ali Awan and Stefan M. Petters

Outline of the talk

- Multicores and Real-time systems
- Sources of Unpredictability
- Caches, Buses, Memory
- Hardware prefetching
- System Management Interrupts, translation look-aside buffers
- Discussions

Real-Time Embedded Systems



A real-time system is a system that reacts to events in the environment by performing predefined actions **within specified time intervals**

Correctness of results depends on value

and its time of delivery

"Time is of the essence"

Real time applications

Expected to exhibit the required behavior within **time bounds.**

Essential: **upper bound** on the execution times of all tasks known at design time

Commonly called the Worst-Case Execution Time (WCET) and depends on the environment in which the application is executed

Platform: Earlier deployed on uniprocessors

Increasing demands of applications → Deployed on multicores

COTS-based Multicores for hard real time systems: Boon or bane?

- Increased computational power
- SWaP (Lesser Size, Weight, Low Power consumption)
- Natural fault containment for applications
- Decrease in number of computational nodes and wires
- Cost efficient
- Faster time to market
- Integrated functionality → Lesser number of components that can fail (e.g connectors)

COTS: **C**ommercially-available **O**ff **T**he **S**helf

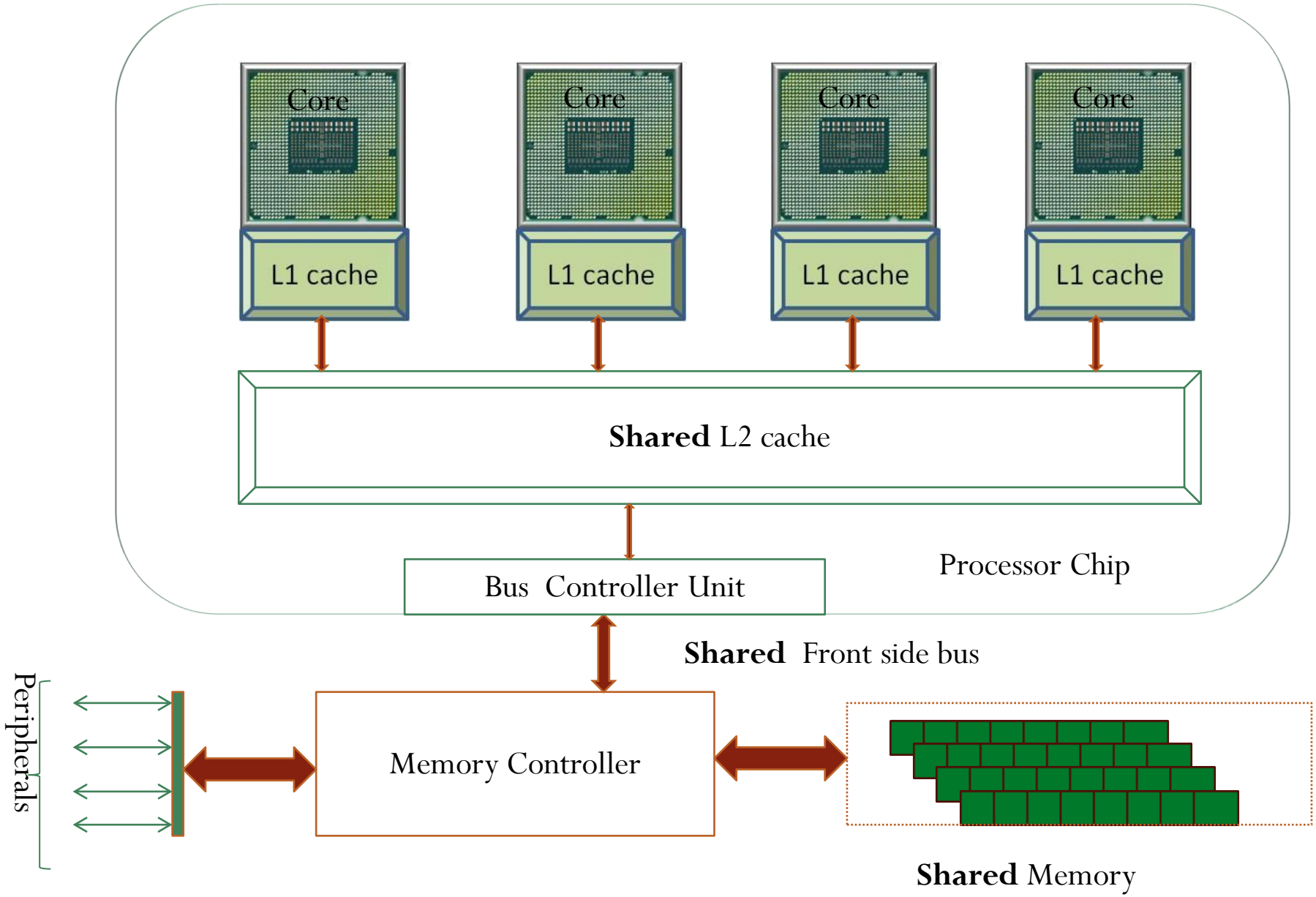
COTS-based Multicores for hard real time systems: Boon or bane ?

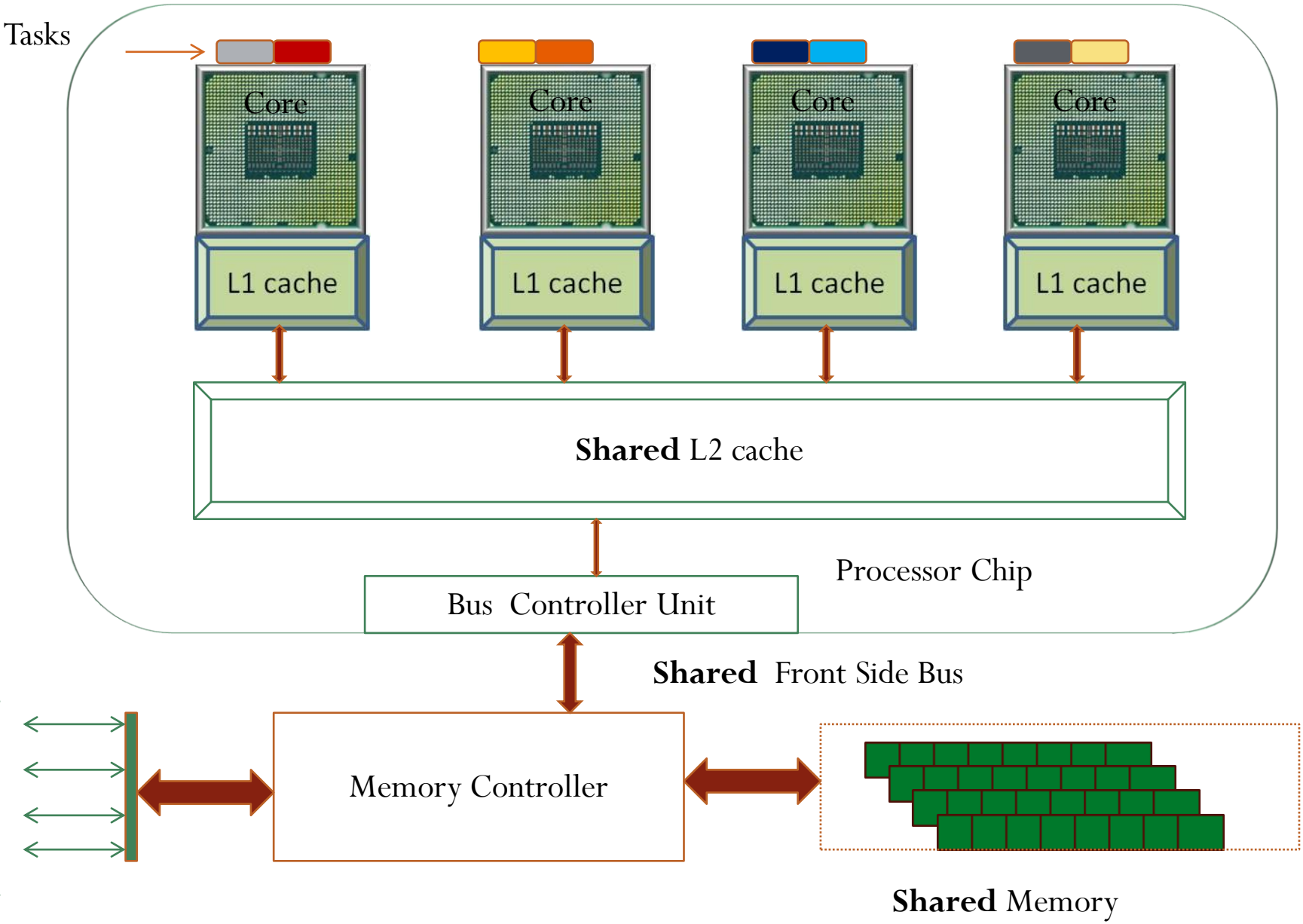
Disadvantages

- Shared low level hardware resources
  - Shared Caches, inter communication channel, shared memory
- No spatial and temporal isolation
  - Variability in execution times depending on the coscheduled tasks
- Complex features make timing analysis challenging
- Increased unpredictability makes certification a nightmare!

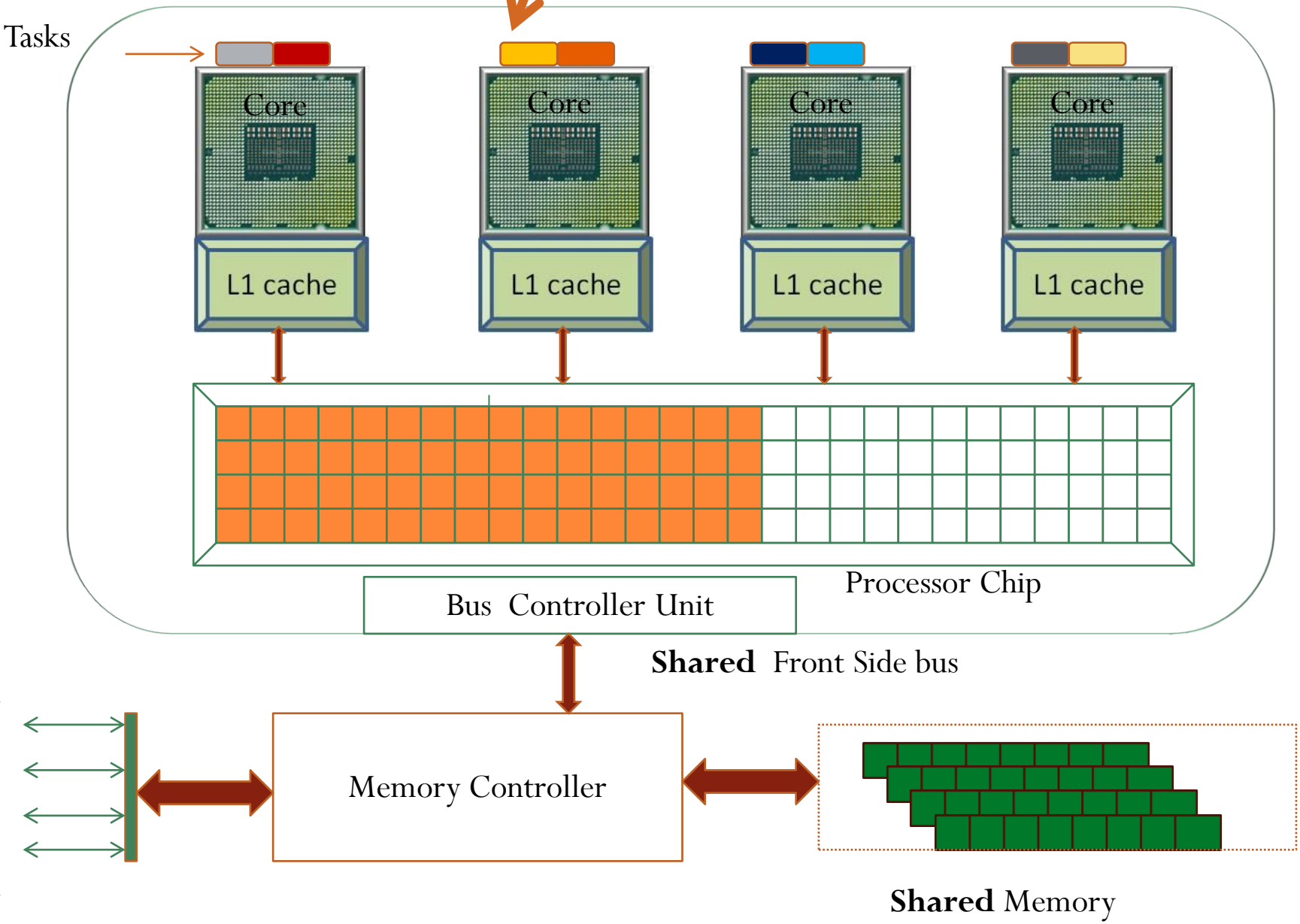
The hard truths:

- Multicores are not yet hard-real time ready
- Safety critical systems form a small market segment.
- Not enough incentives to drive the industry to build "predictable-by-design" multicore systems
- Although around since 2004, no multicore systems are fully certifiable
- The current solution: **Disable all the cores except one!!!!**

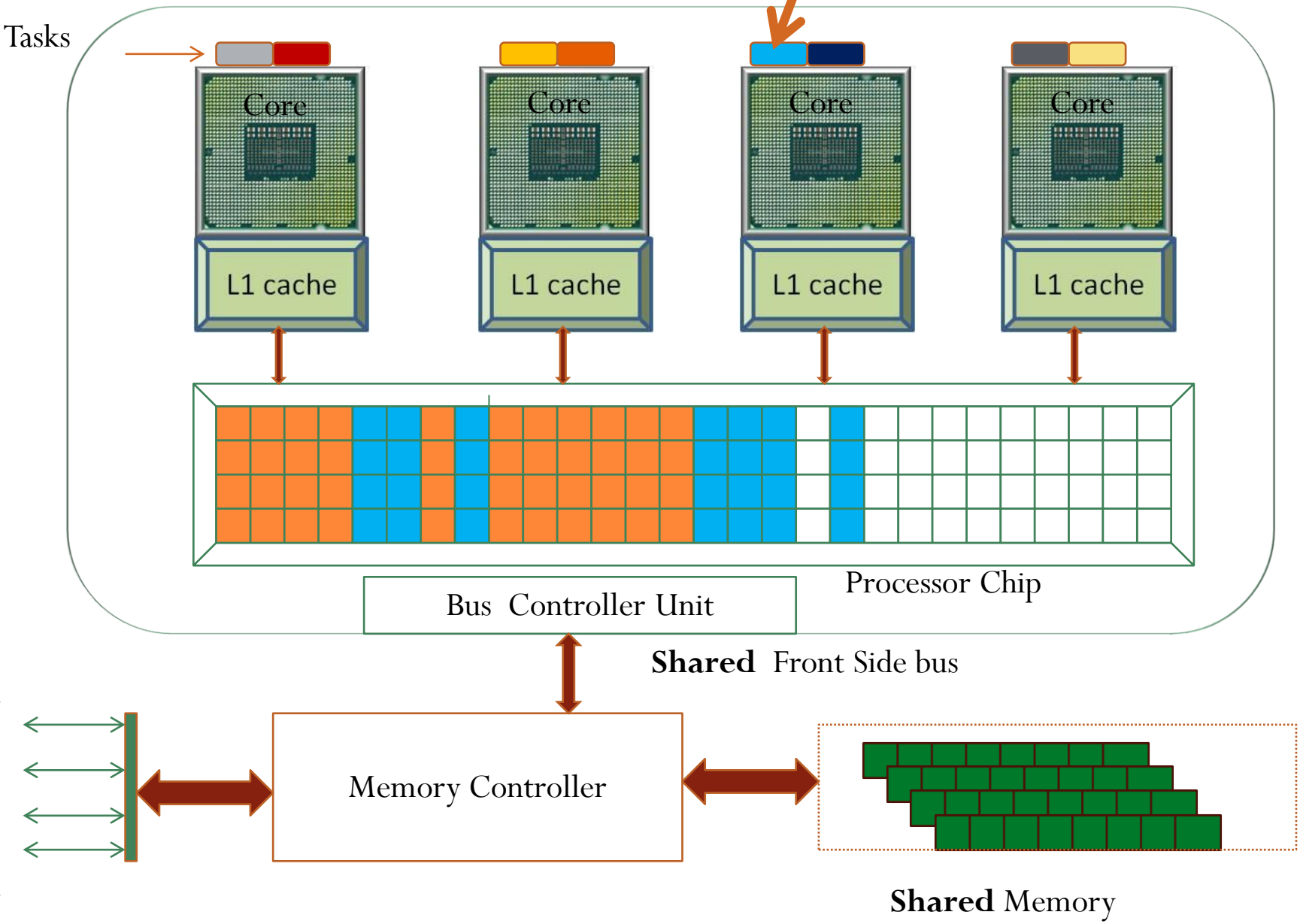

Core
Core
Core
Core
L1 cache
L1 cache
L1 cache
L1 cache
Shared L2 cache
Bus Controller Unit
Processor Chip
Shared Front side bus
Peripherals
Memory Controller
Shared Memory


Tasks
Core
Core
Core
Core
L1 cache
L1 cache
L1 cache
L1 cache
Shared L2 cache
Processor Chip
Bus Controller Unit
Shared Front Side Bus
Peripherals
Memory Controller
Shared Memory

Cache Contention



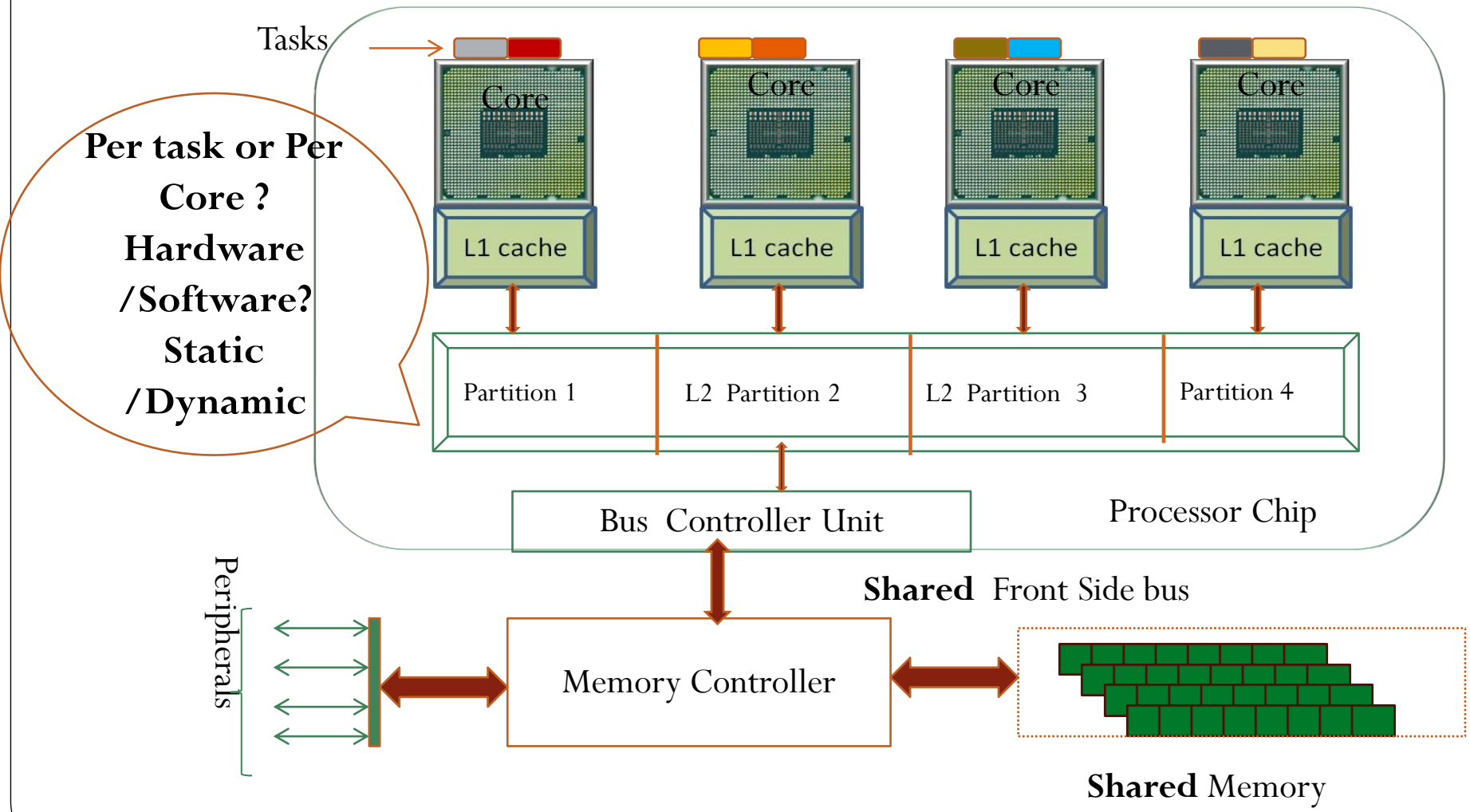
Cache Contention

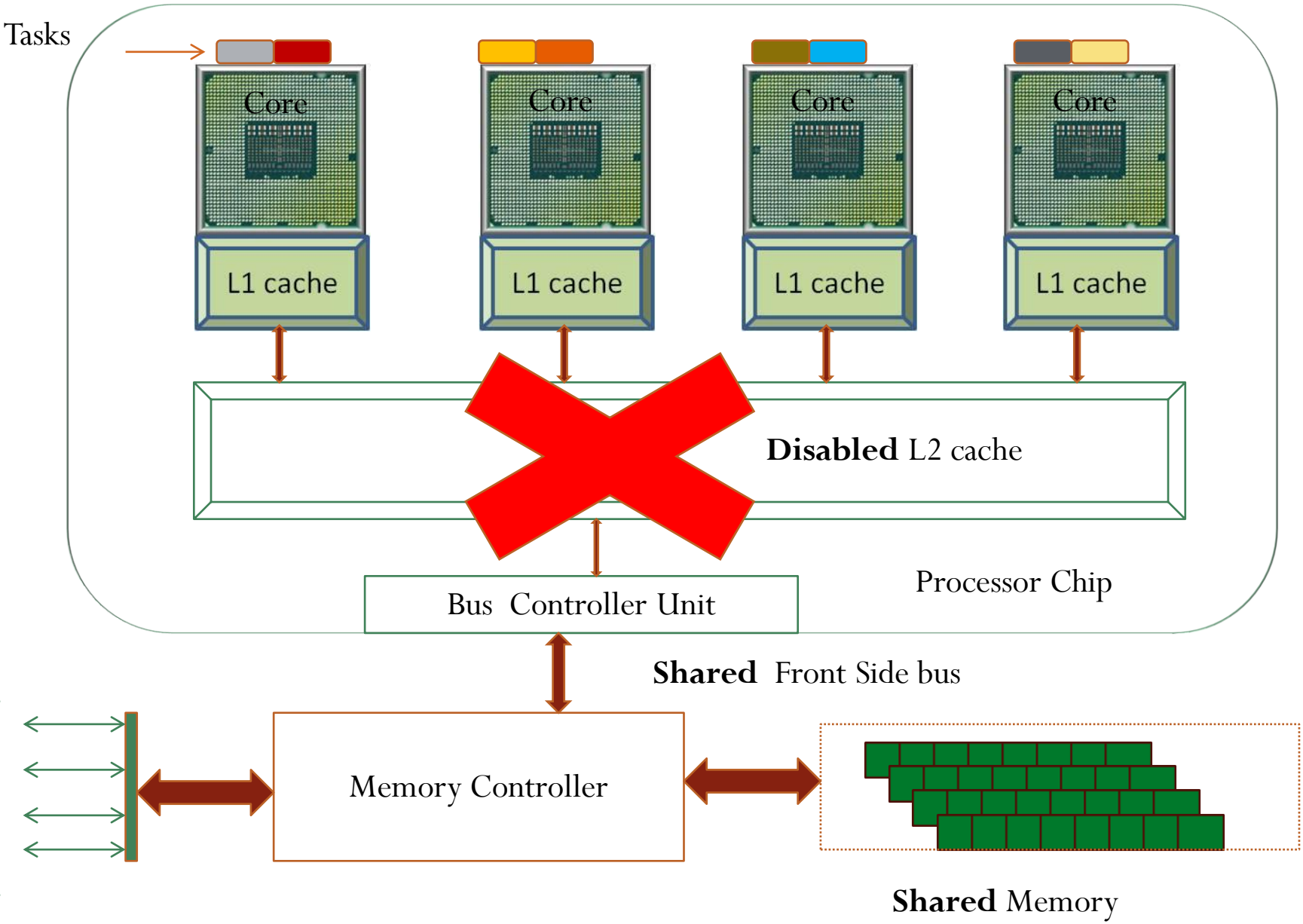


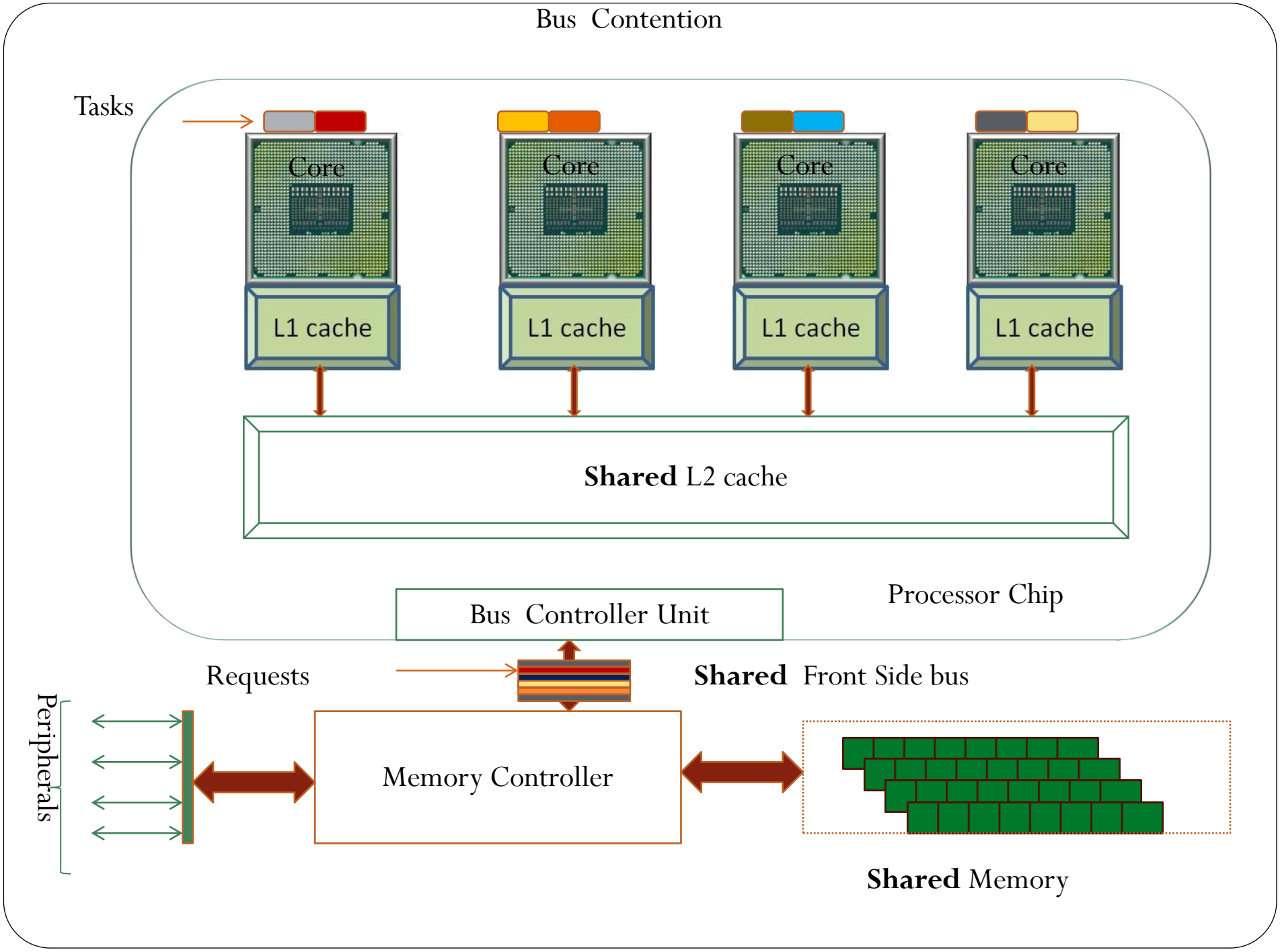
Unpredicatability in Caches

Feature	Types	Timing Analyzability	Modern Multicores
Replacement Mechanism	LRU, PLRU, Random, FIFO	LRU	Generally PLRU
Write policy	Write back, write through	Write through	Generally Write back
Associativity	High	Difficult to analyze	High
Cache ownership	Private /Shared Caches	Shared caches complex to analyze	Very few have private caches

Avoid Sharing : Cache Partitioning




Tasks
Core
Core
Core
Core
L1 cache
L1 cache
L1 cache
L1 cache
Disabled L2 cache
Bus Controller Unit
Processor Chip
Shared Front Side bus
Peripherals
Memory Controller
Shared Memory


Bus Contention
Tasks
Core
Core
Core
Core
L1 cache
L1 cache
L1 cache
L1 cache
Shared L2 cache
Processor Chip
Bus Controller Unit
Requests
Shared Front Side bus
Peripherals
Memory Controller
Shared Memory

Challenges for Bus Contention Delay Analysis of a task

- Based on memory traffic (cache misses) from tasks on other cores
- Bus Requests not tagged with task priorities
- Reordering of request servicing (Higher priority tasks are stalled longer)
- Arbitration policy decides the availability of the bus
- Prefetching adds to the bus traffic

Bus Arbitration policies : TDMA

Time Division Multiple Access Bus

- Real time friendly (Predictable, Composable)
- Non work-conserving
- But generally not used in COTS (Commercially available off the shelf) systems

Assignment of bus time slots

Unpredictability in Buses

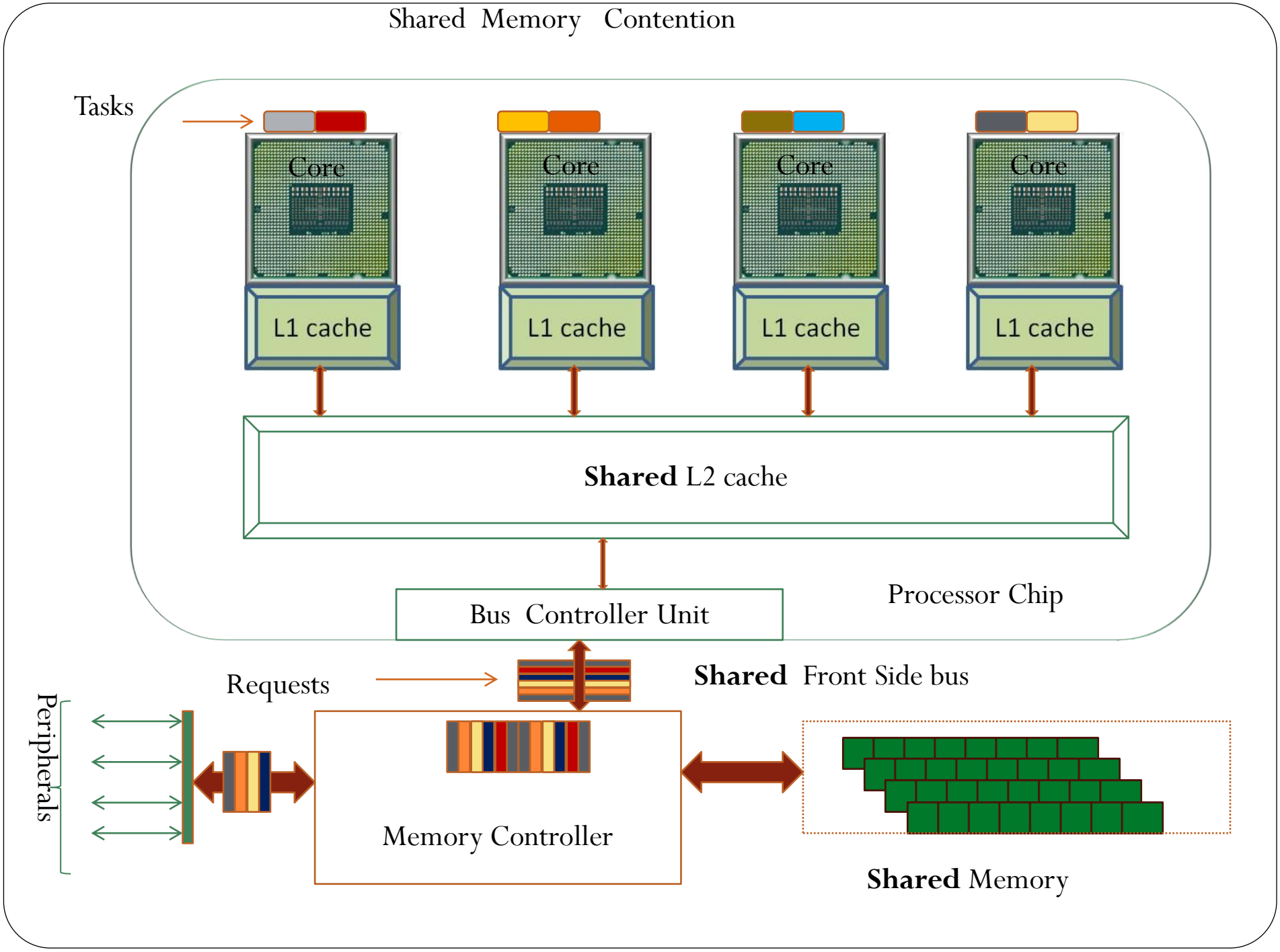
Feature	Types	Prefered for Timing Analyzability	Present in Modern Multicores
Arbitration Mechanism	TDMA, Round Robin, Priority Based, FIFO	TDMA good for Real Time systems	Some weighted round robin/propriety mechanisms
Transaction ordering	Simple Inorder, pipelining, split transaction	Easier with simple Inorder	Split transaction, request reordering Based on internal priorities
Hardware prefetching	Enabled /Disabled	Disabled prefetching, reduces speculations	Enabled by default

Prefetching

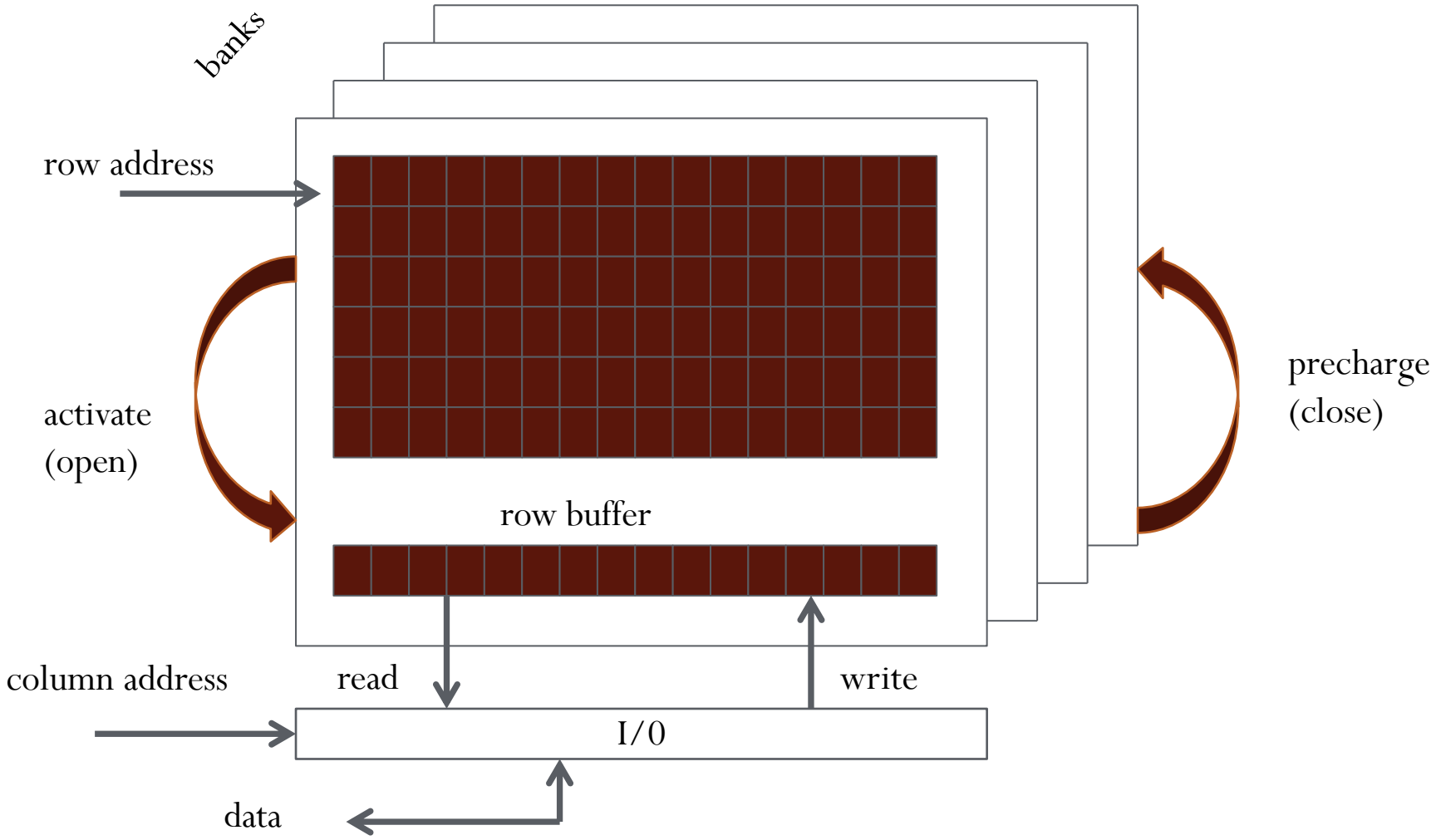
- Prefetch instructions : Software
  - Compile-time analysis, schedule fetch instructions within user program
- Hardware based prefetching
  - Intel:
    - Adjacent cache line prefetcher
    - Hardware Prefetching : Detecting a stride in the array access pattern
- Disadvantages of OS transparent prefetching
  - Cache pollution: Replaces key cache blocks
  - Uses bus bandwidth
  - Delays important requests
  - Timing unknown to user and causes variations in program behavior

Some interesting research ideas

- Predictable execution model
  - Compatible and predictable execution phases --prefetch all the required data in your time slot and then start executing (assumes TDMA)
  - Controlled data acquistion and replication models proposed
- Profiling task behavior to analyze requests patterns for single task, multiple tasks and at the core levels and analyze the interference on the bus
- Table driven bus arbiters -- different bus access schedules based on applications
- Budgeting the bus bandwidth --Memory centric scheduling
- And more in the paper


Shared Memory Contention
Tasks
Core
Core
Core
Core
L1 cache
L1 cache
L1 cache
L1 cache
Shared L2 cache
Bus Controller Unit
Processor Chip
Requests
Shared Front Side bus
Peripherals
Memory Controller
Shared Memory

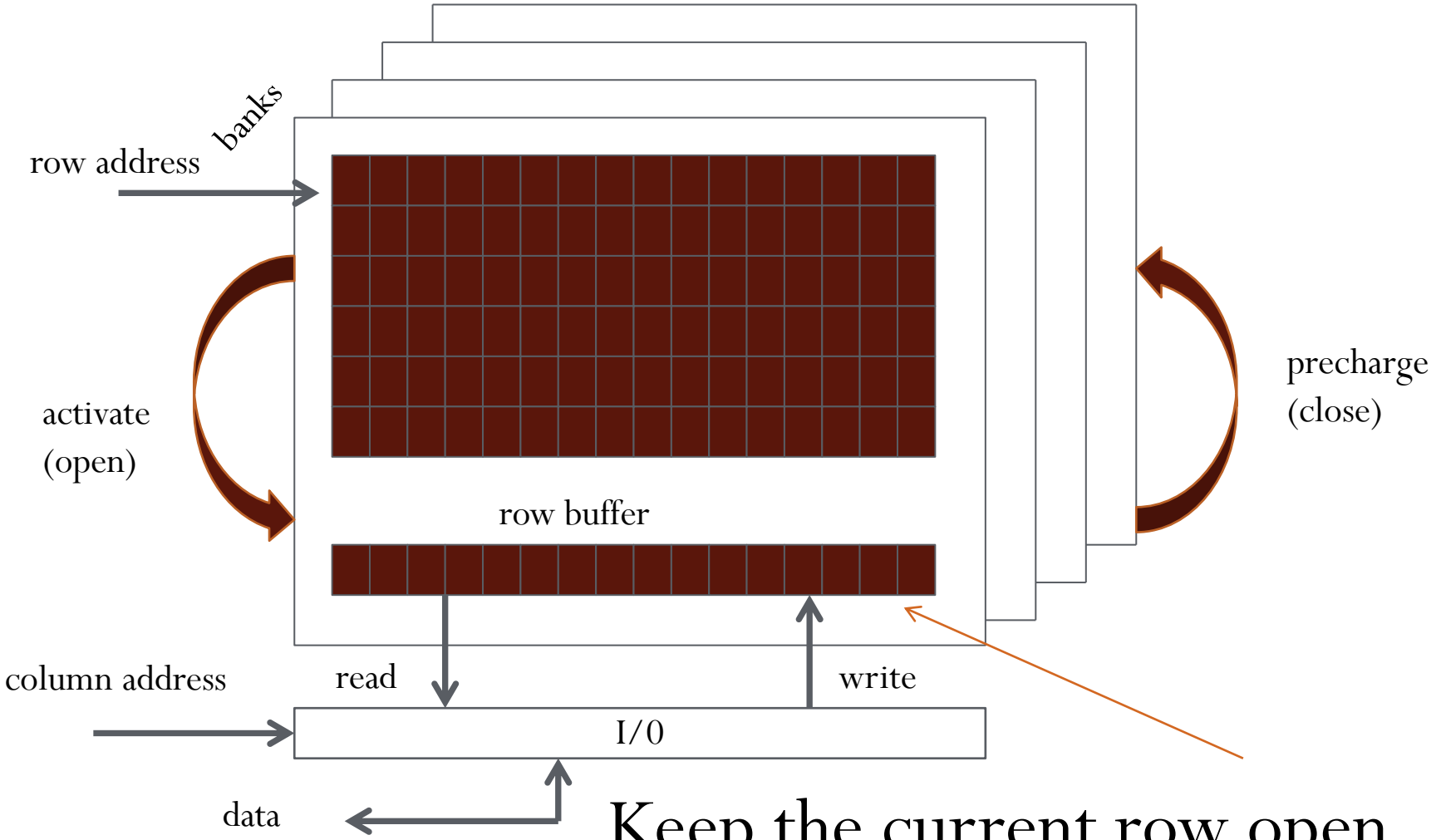
Memory organization



Unpredictability in memory accesses

- DRAM is the center of memory hierarchy:
  - High density and high capacity
  - Low cost but slow access (compared to SRAM)
- DRAM access generally considered to be uniform.
- Non-uniform access latencies exist within DRAM
- The Row-buffer caches the last accessed row in DRAM

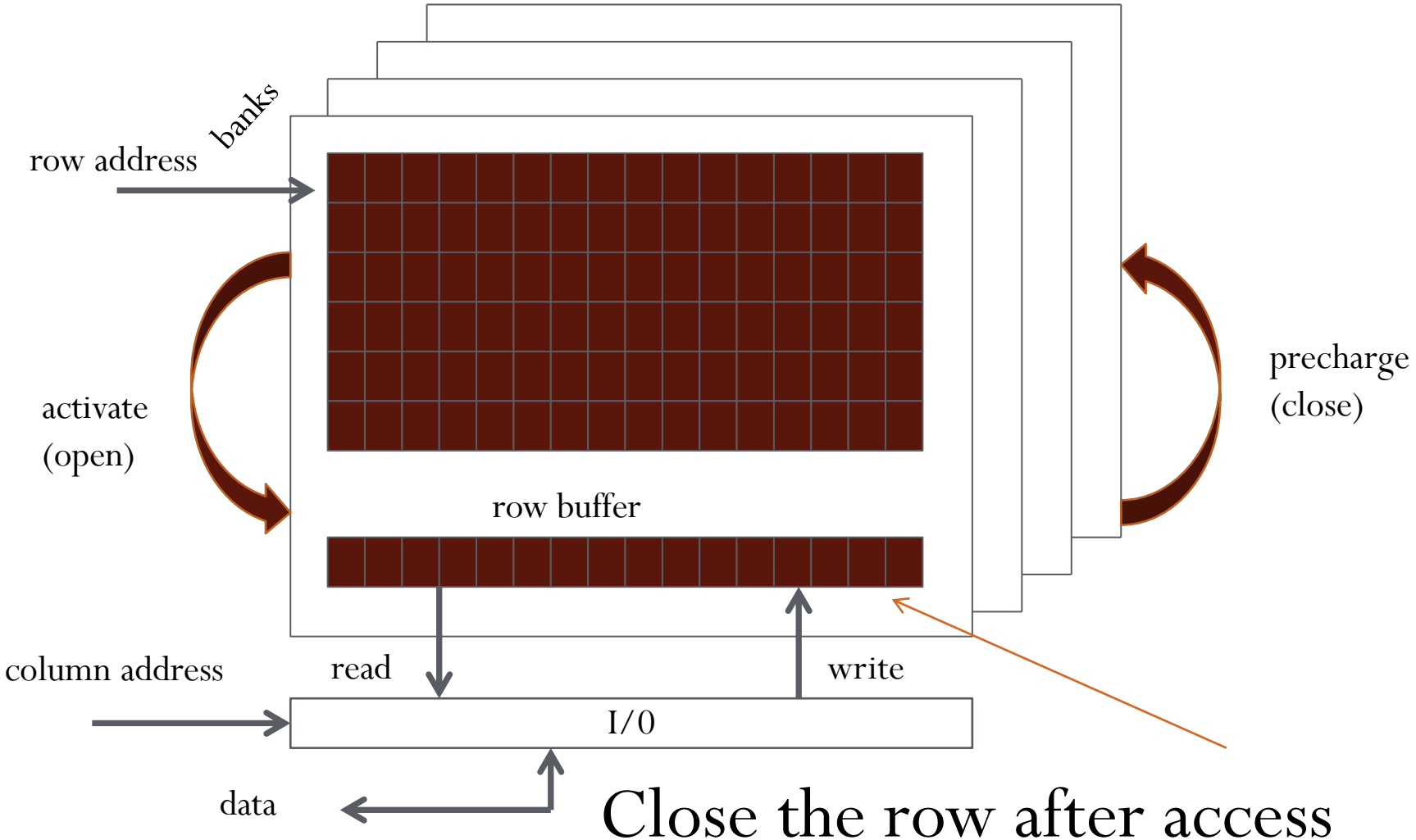
Page policy : Open Page



Keep the current row open

Good if high locality of reference

Page policy : Close Page



Close the row after access

Good for random references

Unpredictability in DRAM Access

Page Policies

- Open page policy
- Close page policy
- Choice of policies -> variability in access times

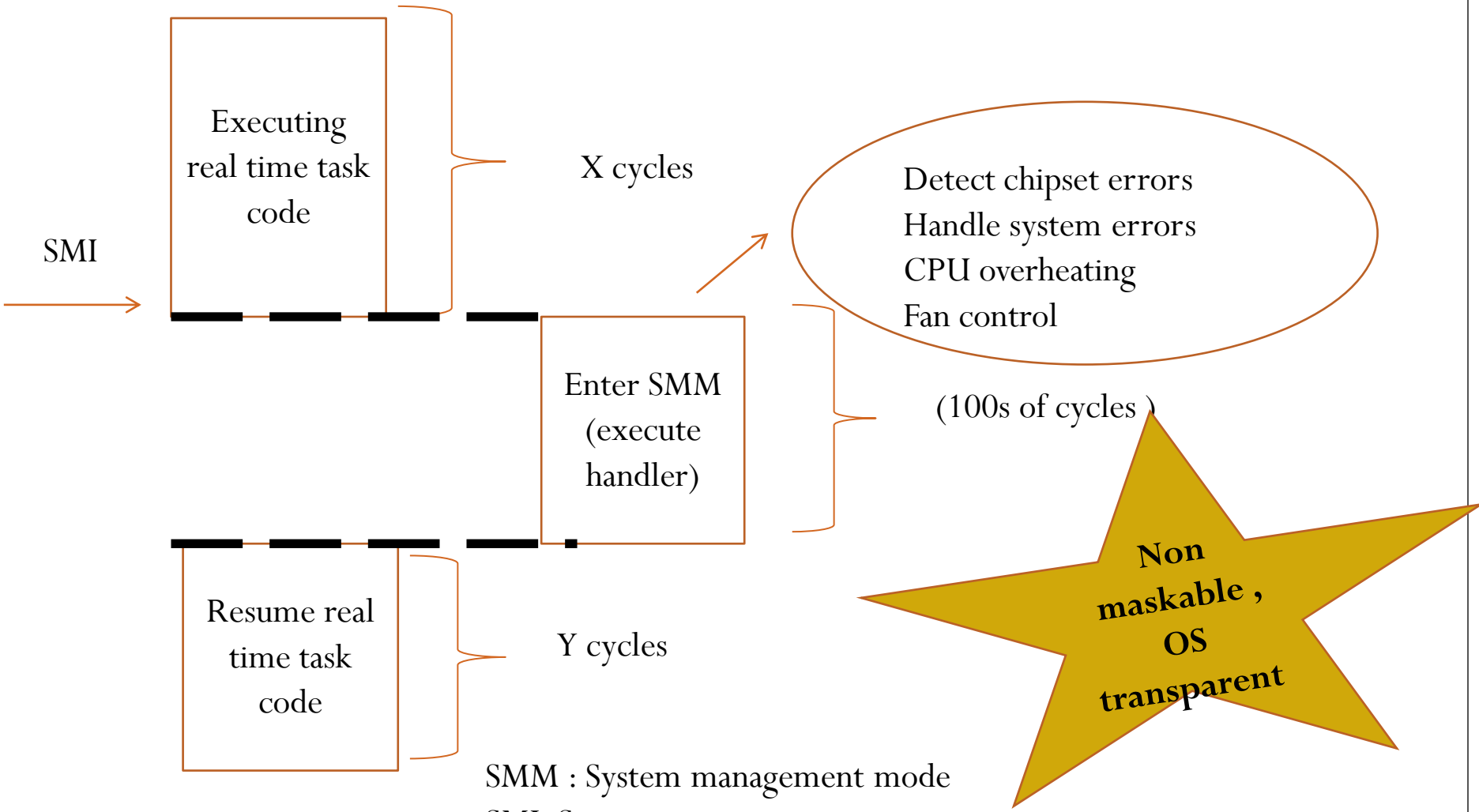
Request reordering

- Prioritize requests that are present in an open row
- Prioritize reads over writes

DRAM refreshes

- Periodic refresh (every 7.5 microsecs)
- Irregular refreshes (in idle cycles)
- Increase the stall

Effect of system management interrupts

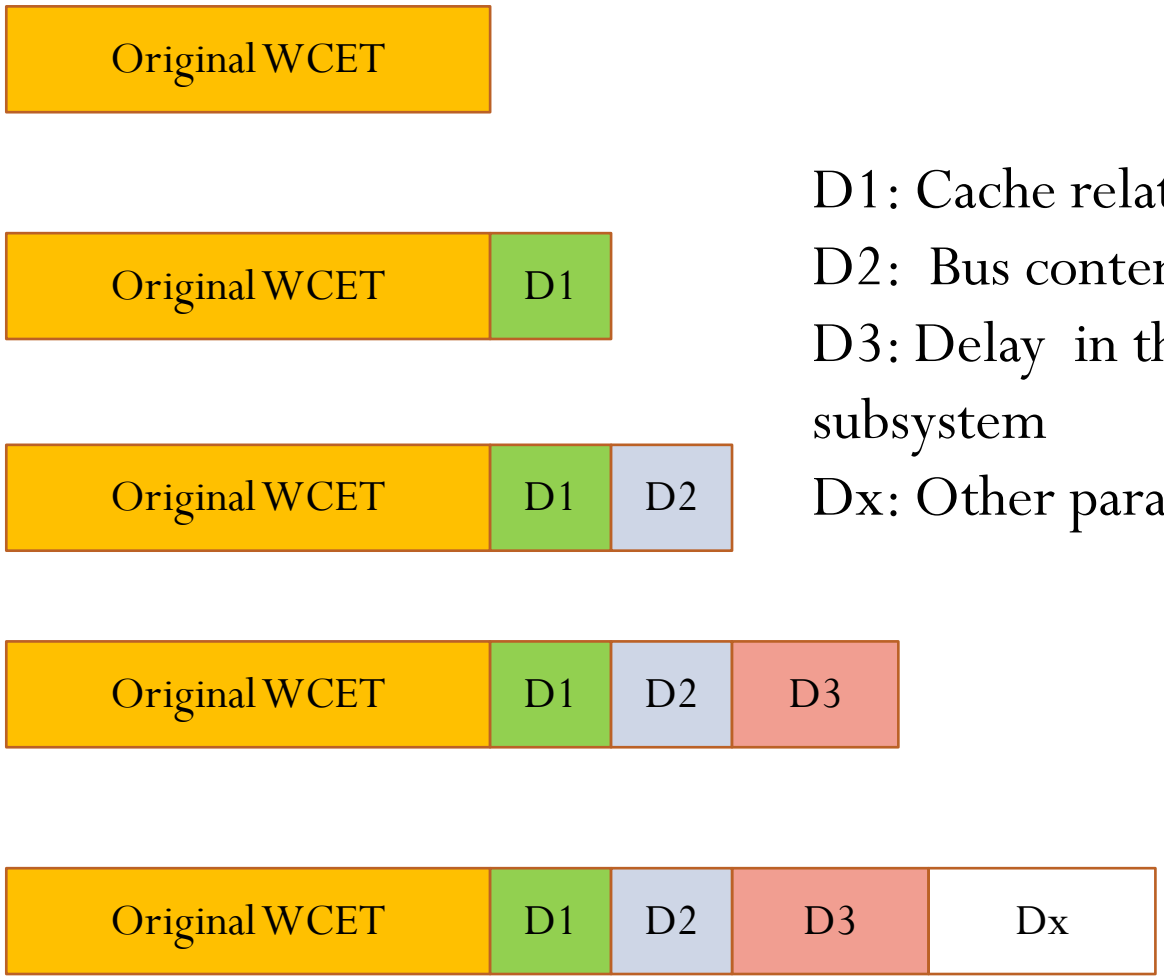


Thermal and Power saving strategies

- Processors like Intel employ "sleep states" by varying the power supplied to different devices
- Wake up latencies vary with different sleep state levels
- Deeper sleep states – Higher wake up latencies (More Idle devices are powered down)
- If a real-time task is now scheduled–it incurs the wake up latency
- Power saving strategies cannot be disabled --may cause harm in the long run
- Adaptive Thermal Monitoring
  - Reduce the temperature when threshold is breached by reducing the frequency and voltage adaptively

Many of these features are enabled by default and must be accounted for in the analysis or be disabled with caution

Variability in the Execution times



D1: Cache related Preemption delay

D2: Bus contention delay

D3: Delay in the shared memory subsystem

Dx: Other parameters

Discussions

- Multicores are here to stay and will be eventually adopted by safety critical systems
- Computational capabilities undisputable but must be assessed for predictability for real time systems
- Unpredictability at each level of system design in current hardware.
- Task memory profiles also an inherent characteristic (apart from the Ci,Ti, Di)
- Holistic end to end analysis must consider all the possible sources of unpredictability to deploy safe and robust systems
- Academic research generally assume simplified models, validated on unrealistic simulators which do not capture the various sources of unpredictability – Pessimistic results and solutions cannot be applied to the industry
- Need to study the hardware in detail to achieve tighter and acceptable bounds

So much more to be done!

Additional details in paper

- Research study in the related areas
  - Different approaches to solve these issues
  - Limitations of current approaches and suggestions