

Multicore Processor Issues for Real-Time Systems

Presenter: Dionisio de Niz



Copyright 2018 Carnegie Mellon University. All Rights Reserved.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

Carnegie Mellon® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

DM18-0745

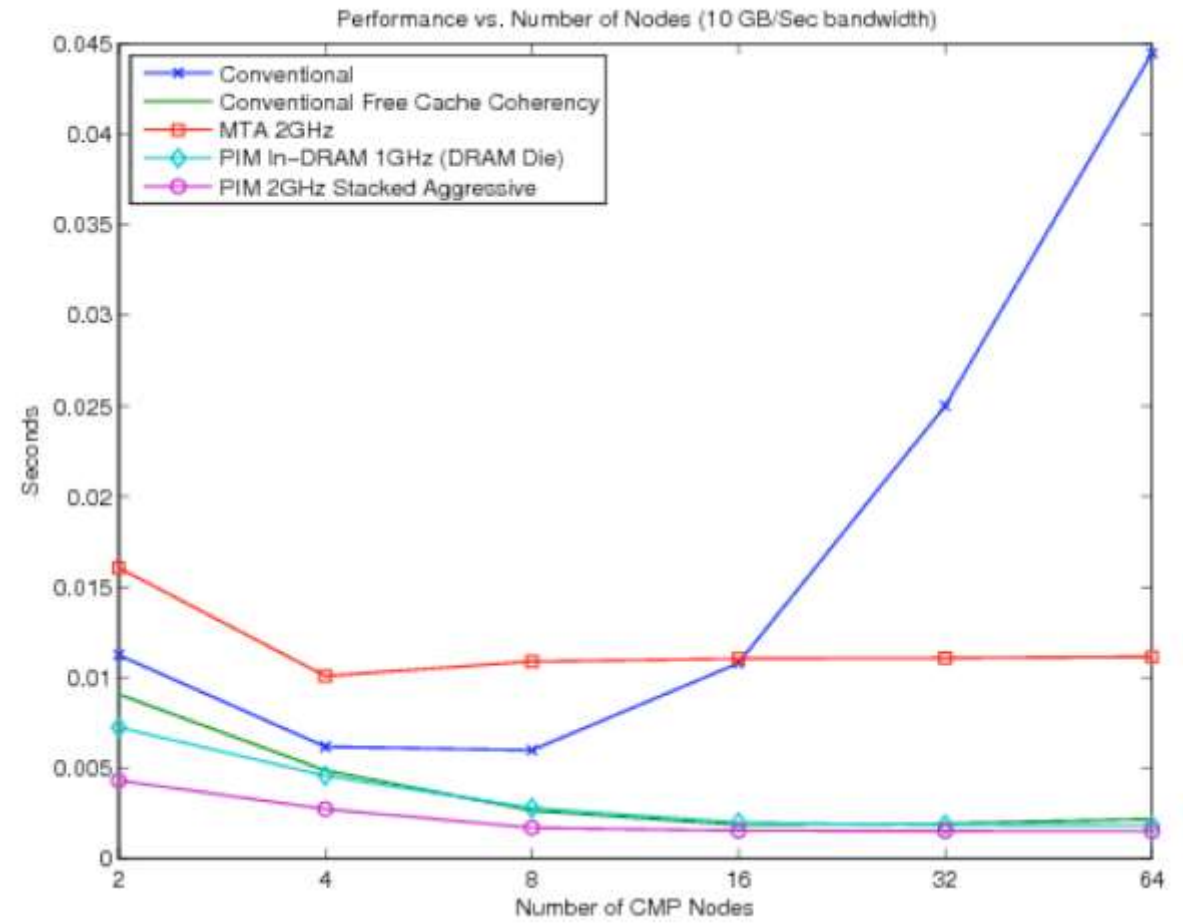
Multicore Becomes Mainstream

Power Wall: May 2004 Intel cancels the Tejas CPU

- Projected to run at 7GHz
- Generated 150 Watt of heat at 2.7GHz
- Intel starts with dual core

Memory Wall: Sandia Labs

- 2007/2008 study on memory bandwidth in data intensive applications
- With conventional memory
 - More than 8 cores performance decreases
 - **More than 16 cores worse than 1 core**



Good News !

Working solutions for the memory wall

Growing number of cores – more computing power!

- June 5, 2018:
 - Intel 28 cores at 5GHz (for Q4)
- June 6, 2018 :
 - AMD Threadripper 2 32 cores 7nm (for Q3)
- NXP QorIQ® P4080/P4040/P4081 4-8 cores 1.5 GHz.
- ARM big.LITTLE
 - 4 big (cortex A15)
 - 4 little (cortex A7)

Bad News! ☹️

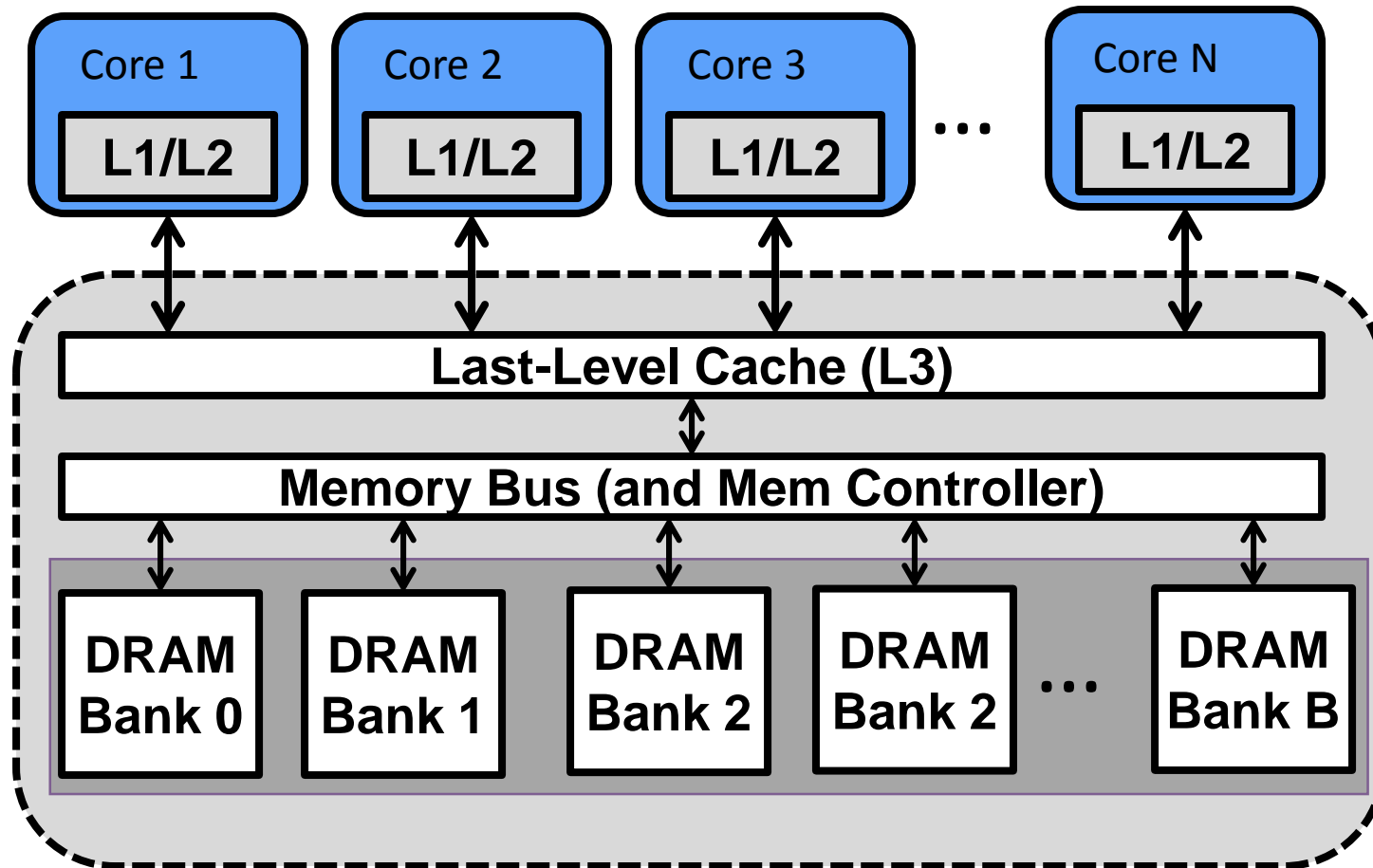
Memory wall solutions focused on throughput

- Can make predictable worst-case execution time (WCET) worst

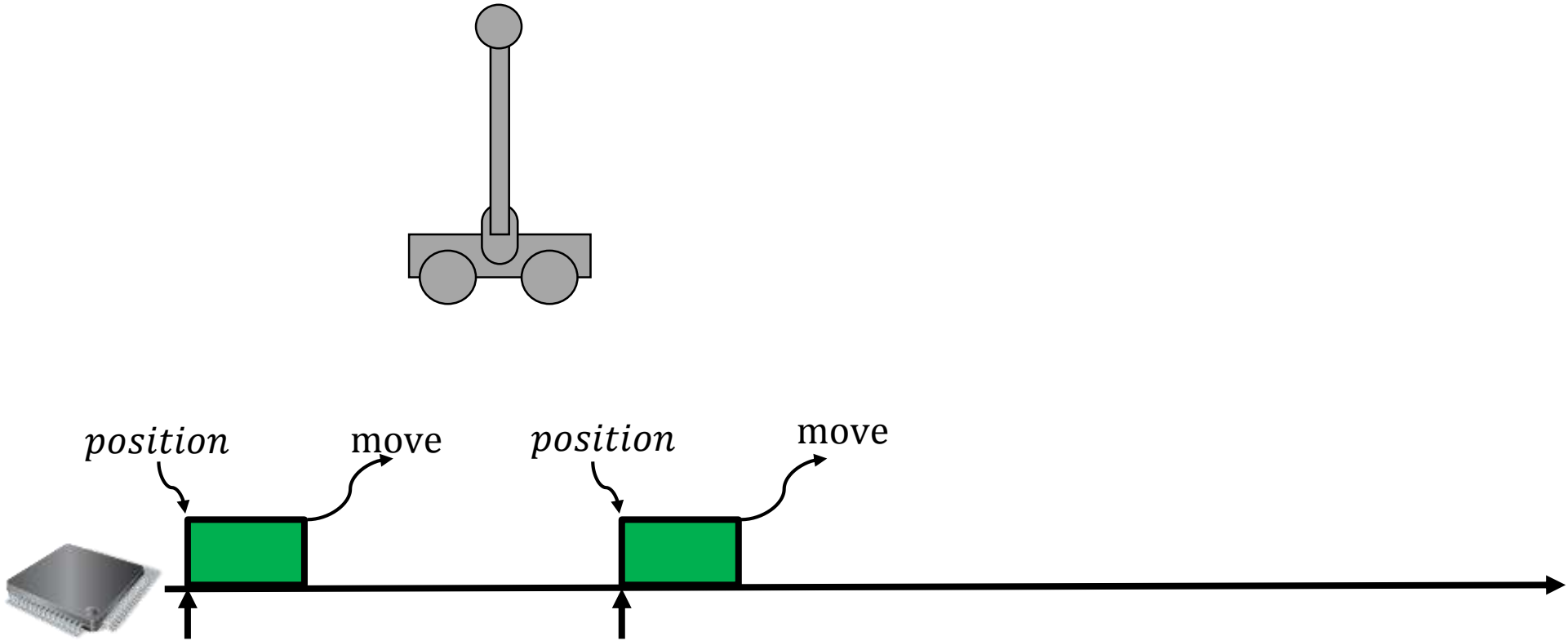
Hardware solutions for throughput hinders timing guarantees / predictability

Hardware solutions for throughput hinders timing analysis

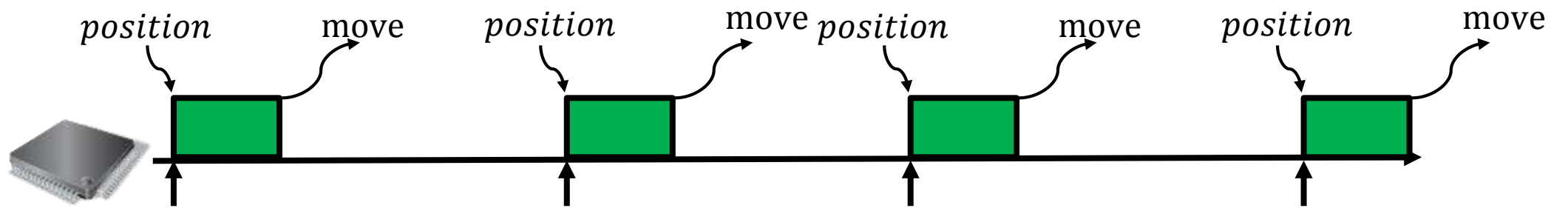
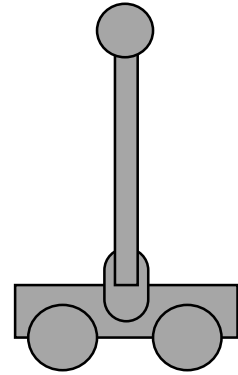
Shared Hardware Resources



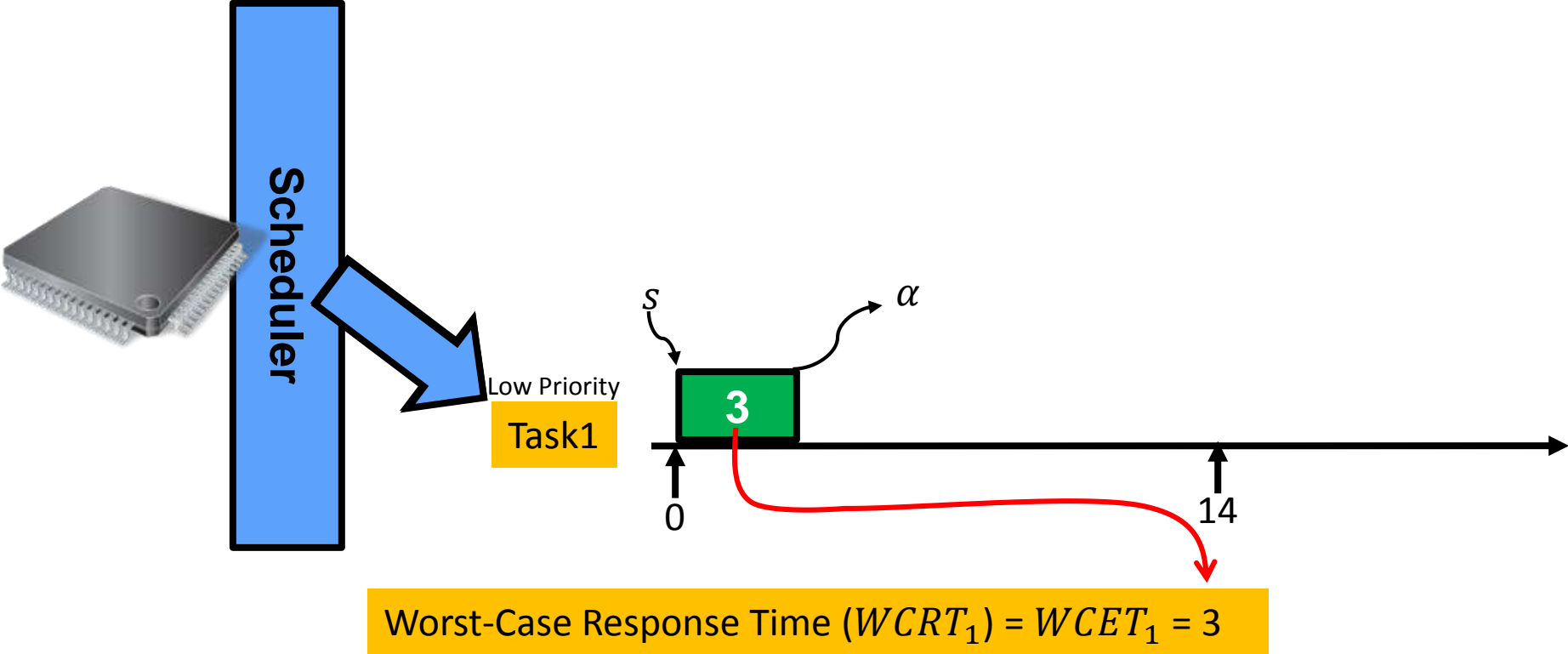
Timing Analysis



Timing Analysis

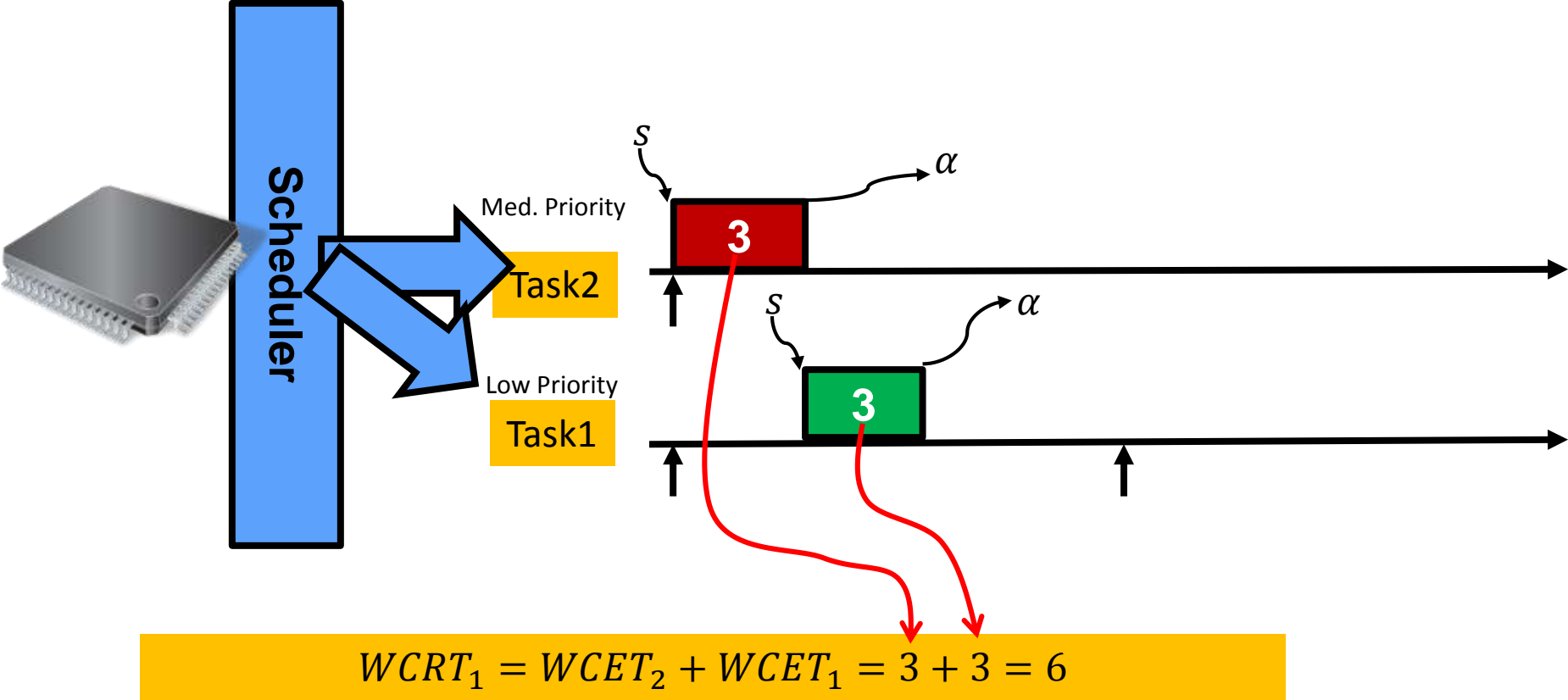


Single-Core Fixed-Priority Scheduling + Rate Monotonic



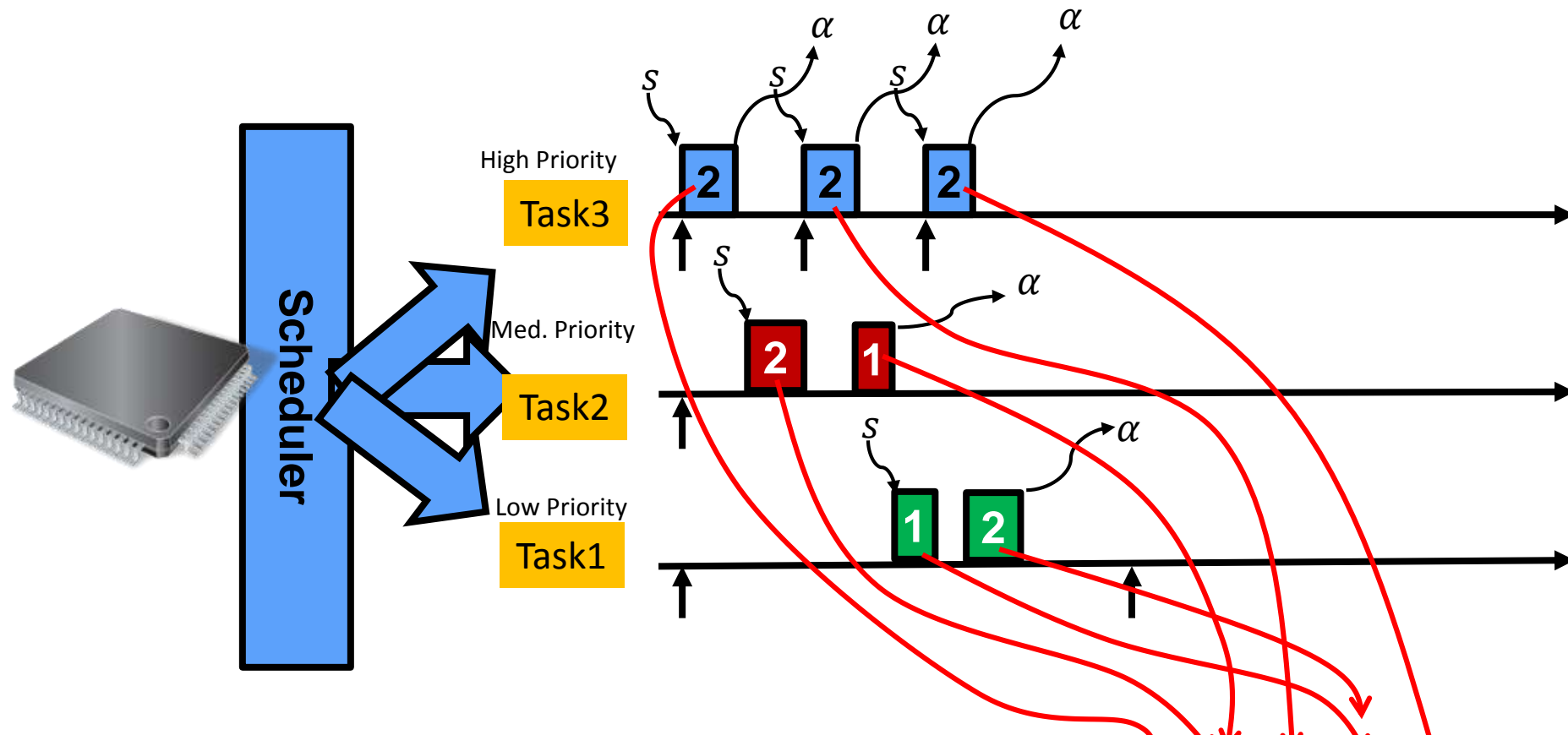
Icons credit: <http://www.doublejdesign.co.uk>

Single-Core Fixed-Priority Scheduling + Rate Monotonic



Icons credit: <http://www.doublejdesign.co.uk>

Single-Core Fixed-Priority Scheduling + Rate Monotonic

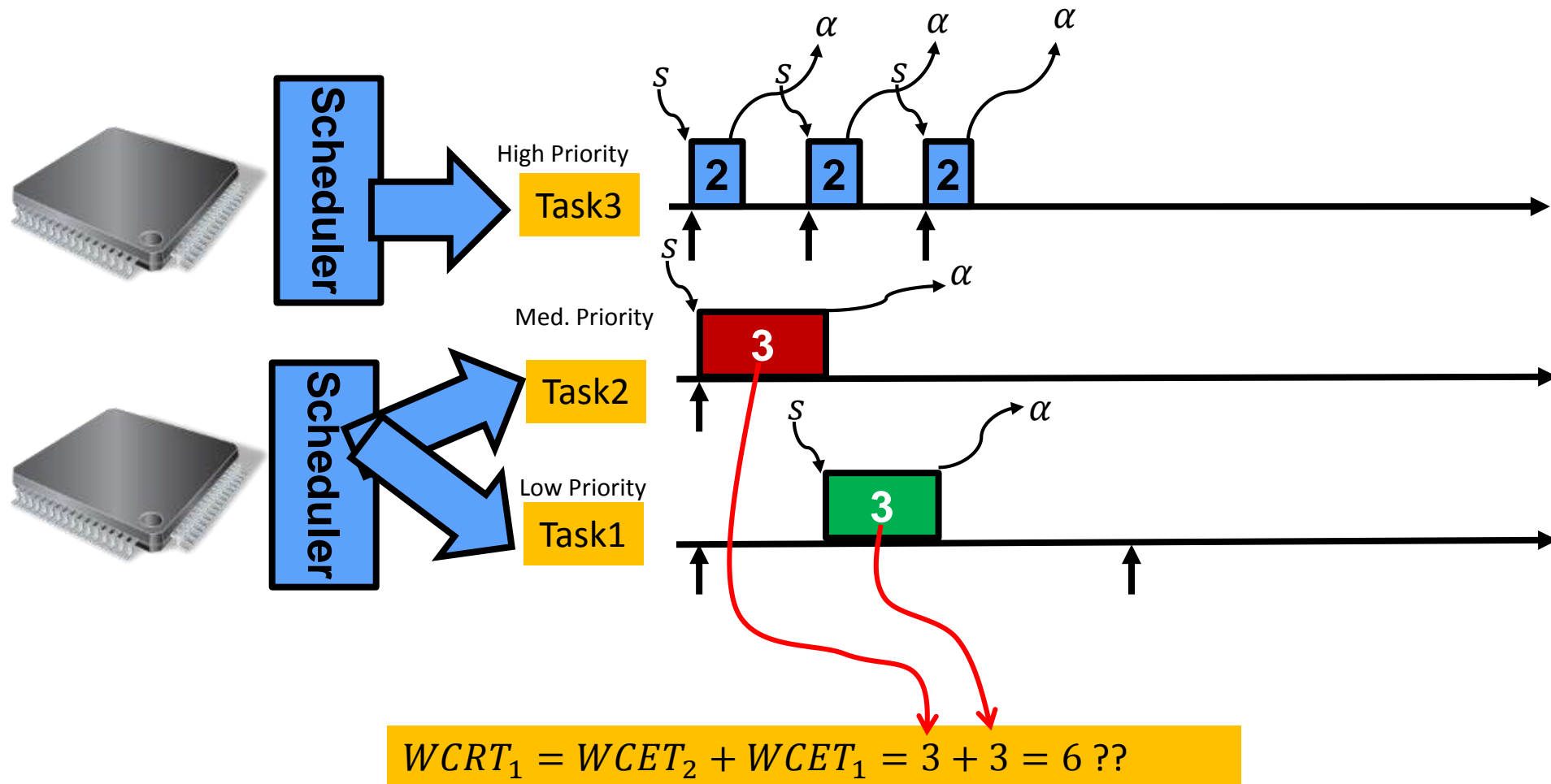


$$WCRT_1 = WCET_3 + WCET_2 + WCET_3 + WCET_1 + WCET_3 = 2 + 3 + 2 + 3 + 2 = 12$$

$$R_i = WCET_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil WCET_j$$

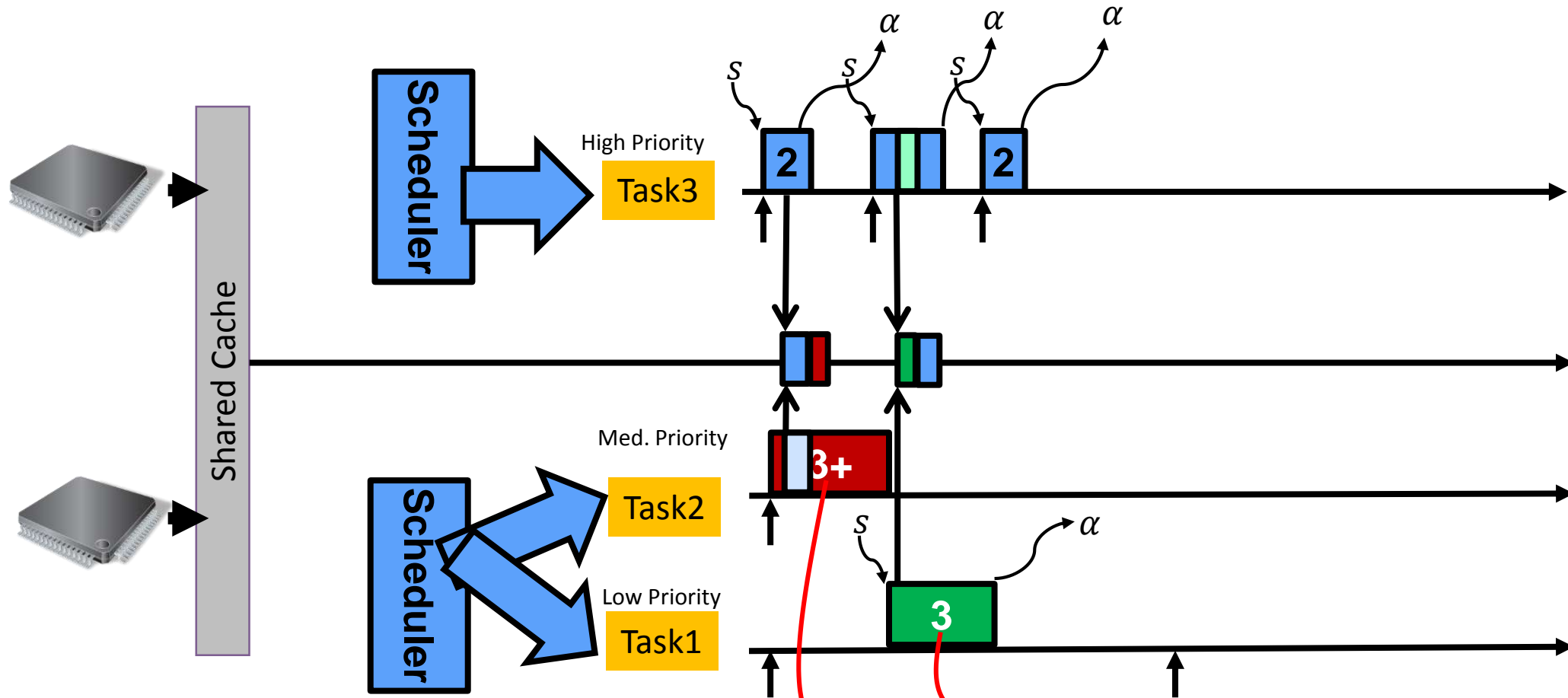
Icons credit: <http://www.doublejdesign.co.uk>

Dual Core: Prevent delays from Task3?



Icons credit: <http://www.doublejdesign.co.uk>

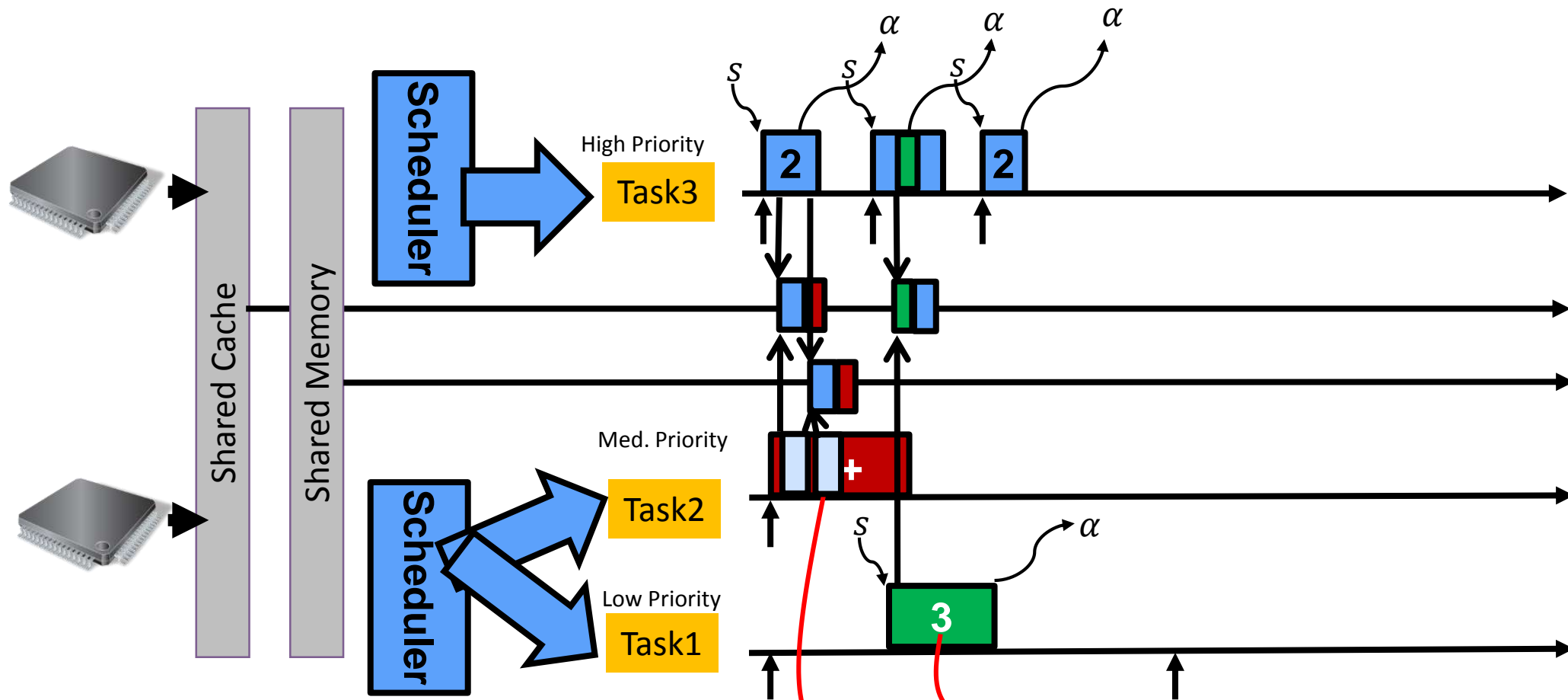
Dual Core: Prevent delays from Task3?



~~$WCRT_1 = WCET_2 + WCET_1 = 3 + 3 = 6 ??$~~

Icons credit: <http://www.doublejdesign.co.uk>

Dual Core: Prevent delays from Task3?



~~$$WCRT_1 = WCET_2 + WCET_1 = 3 + 3 = 6 ??$$~~

~~$$R_i = WCET_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil WCET_j$$~~

Icons credit: <http://www.doublejdesign.co.uk>

Shared Hardware Delays: Shared Cache



Shared Cache

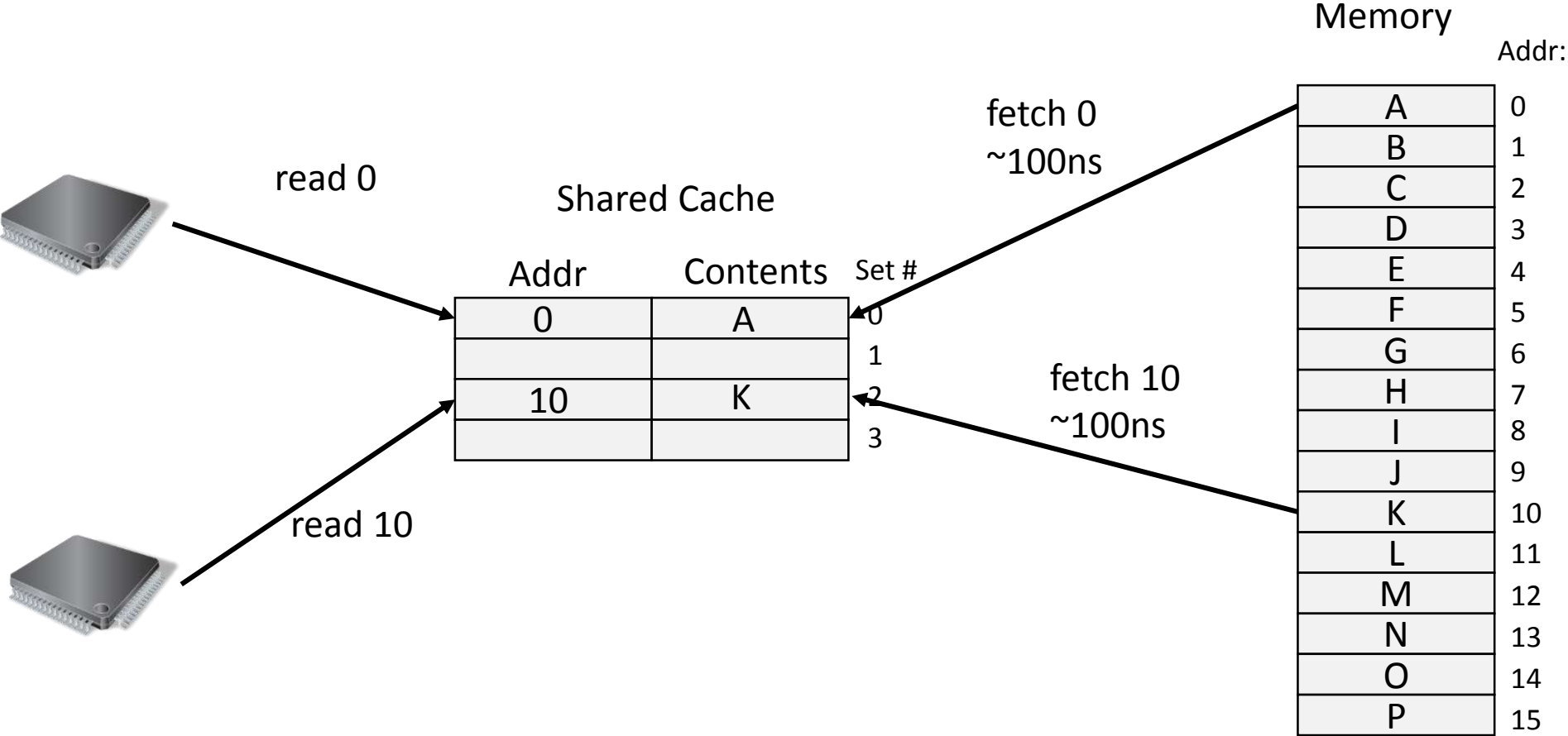
Addr	Contents	Set #
		0
		1
		2
		3

Memory

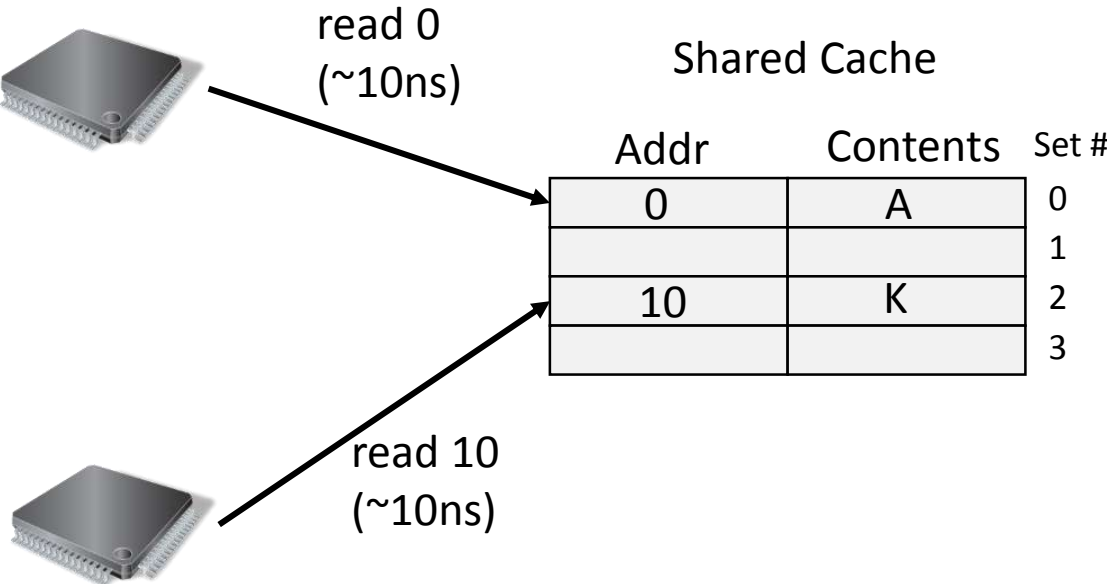
Addr:

A	0
B	1
C	2
D	3
E	4
F	5
G	6
H	7
I	8
J	9
K	10
L	11
M	12
N	13
O	14
P	15

Shared Hardware Delays: Shared Cache



Shared Hardware Delays: Shared Cache

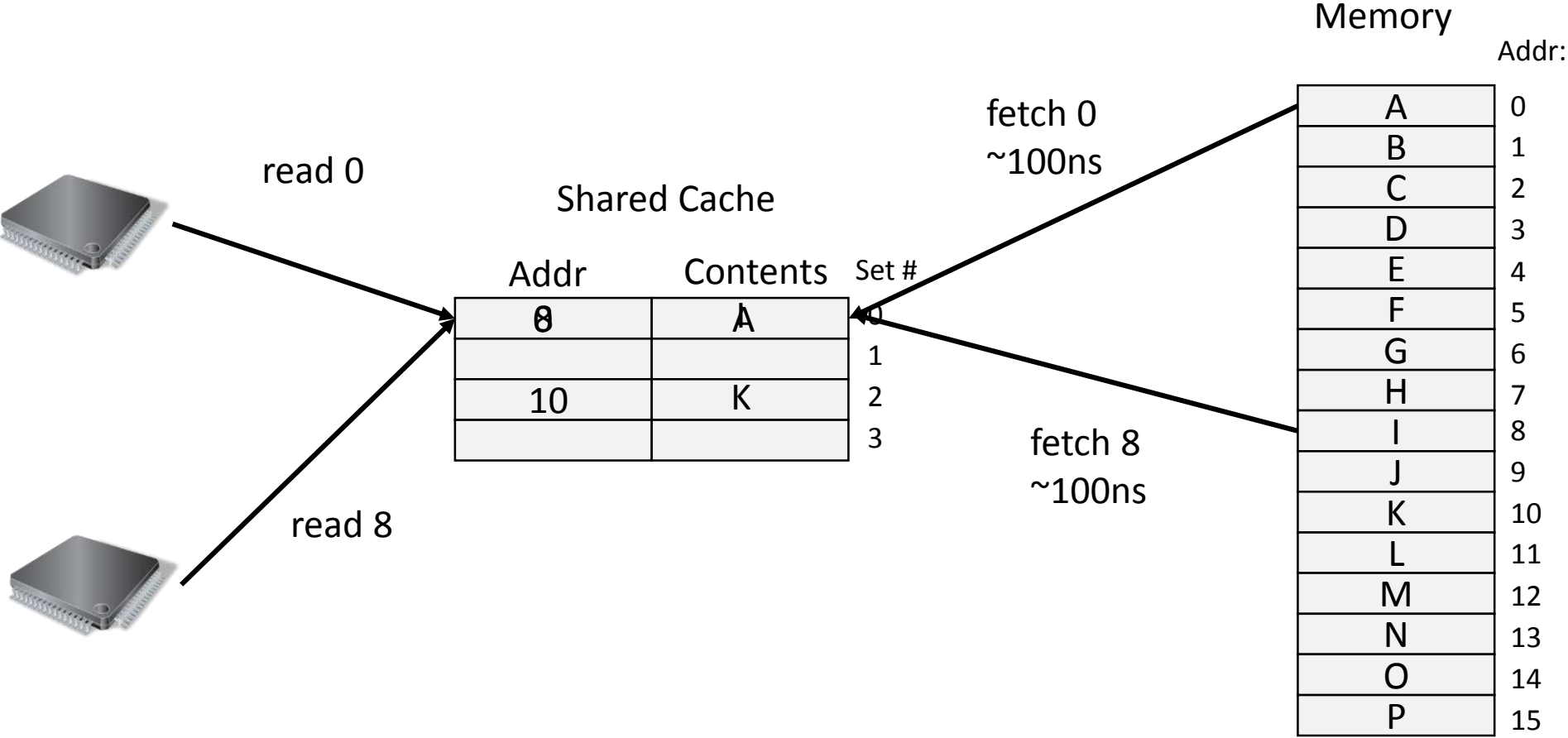


Memory

Addr:

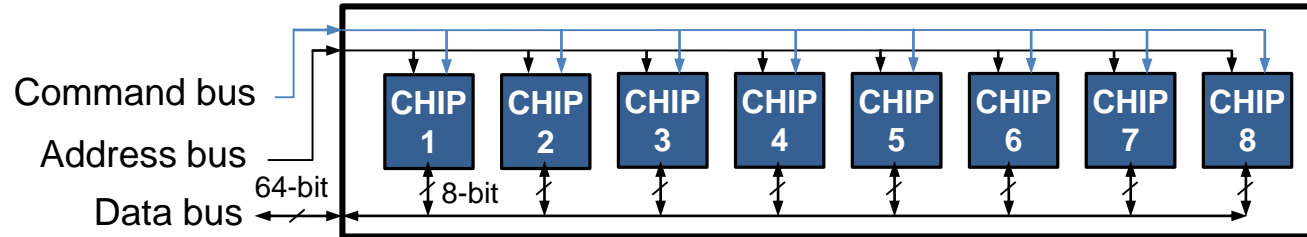
A	0
B	1
C	2
D	3
E	4
F	5
G	6
H	7
I	8
J	9
K	10
L	11
M	12
N	13
O	14
P	15

Shared Hardware Delays: Shared Cache

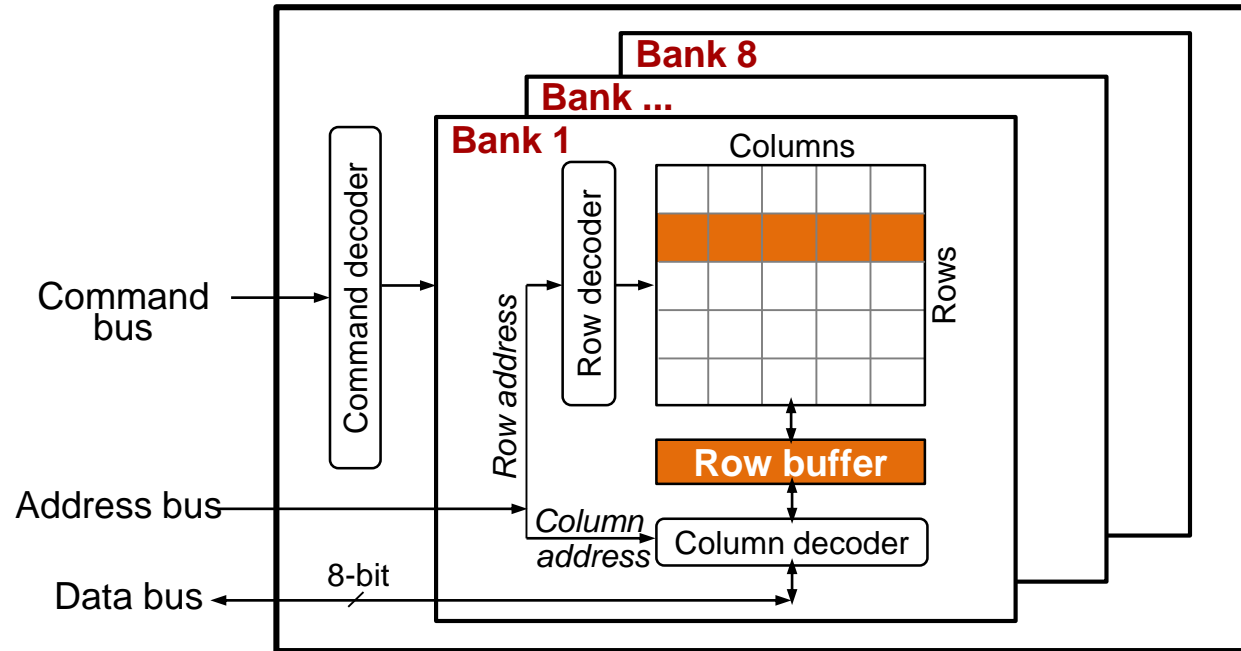


DRAM Memory Organization

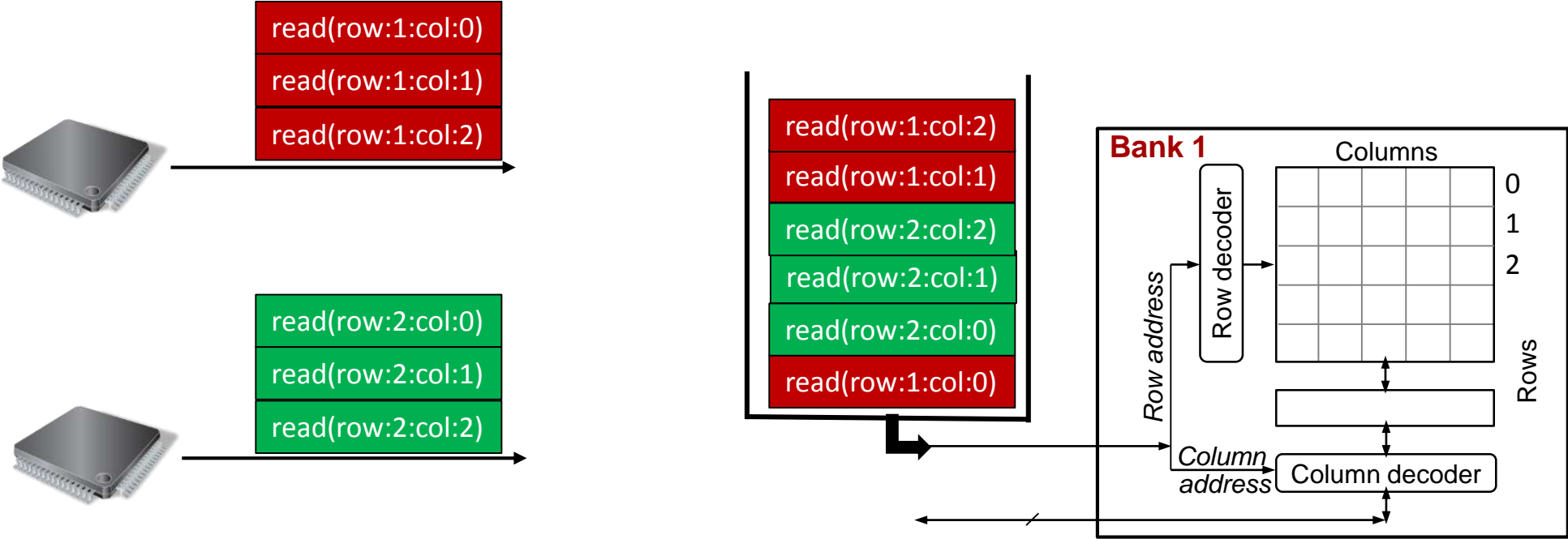
DRAM Rank



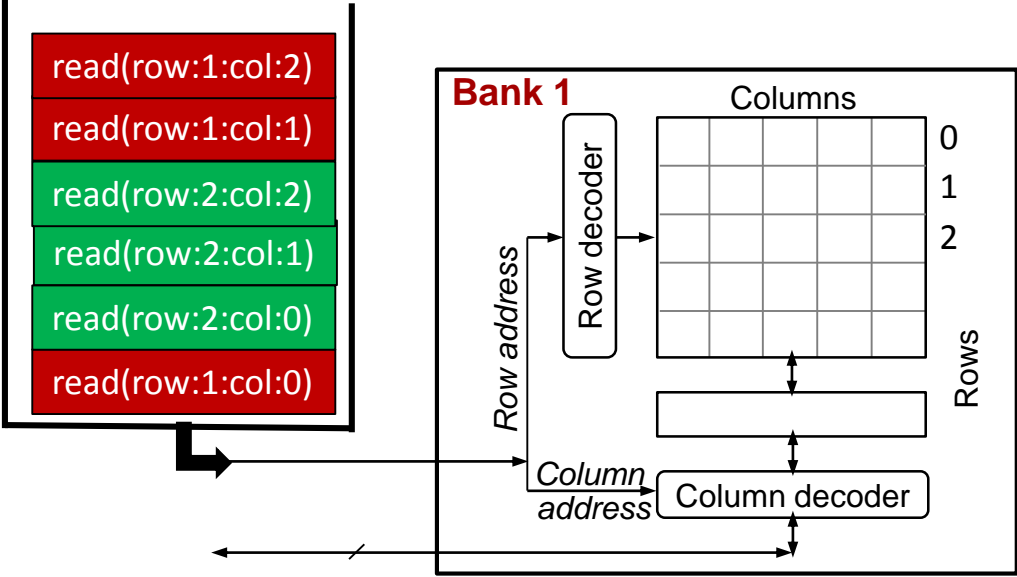
DRAM Chip



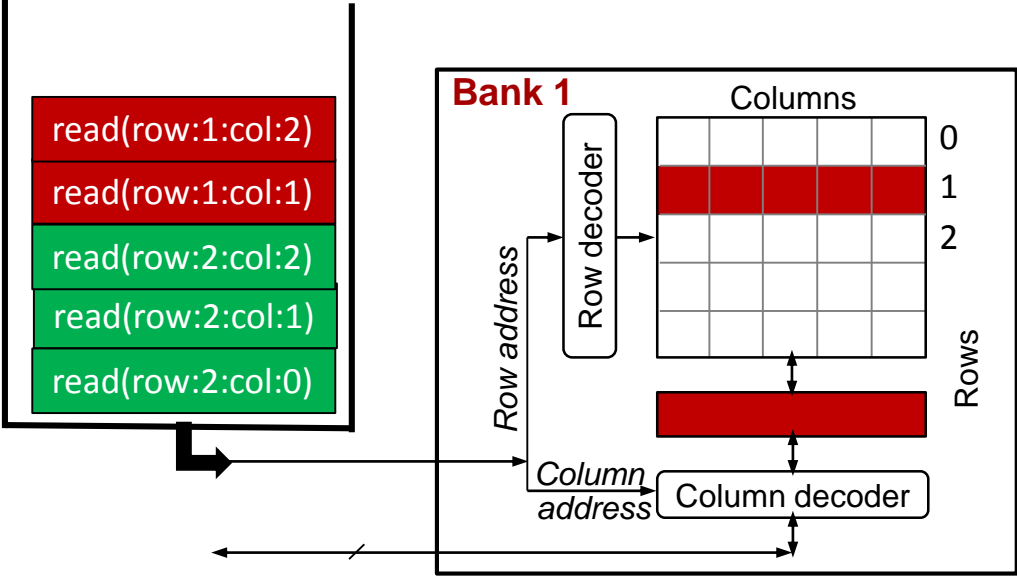
Memory Banks Delays



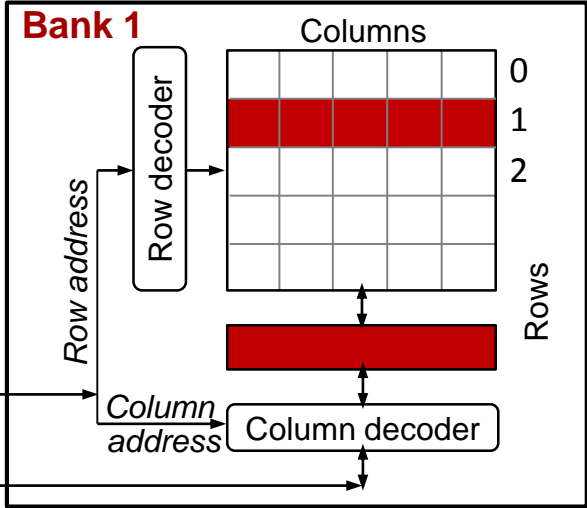
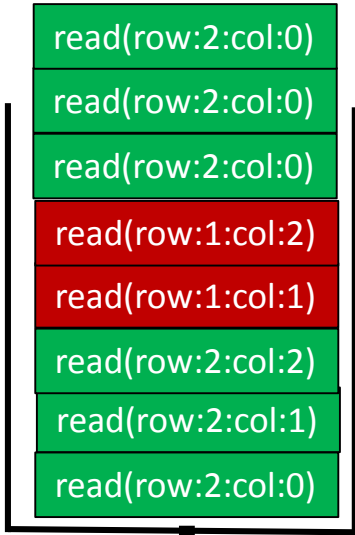
Memory Banks Delays



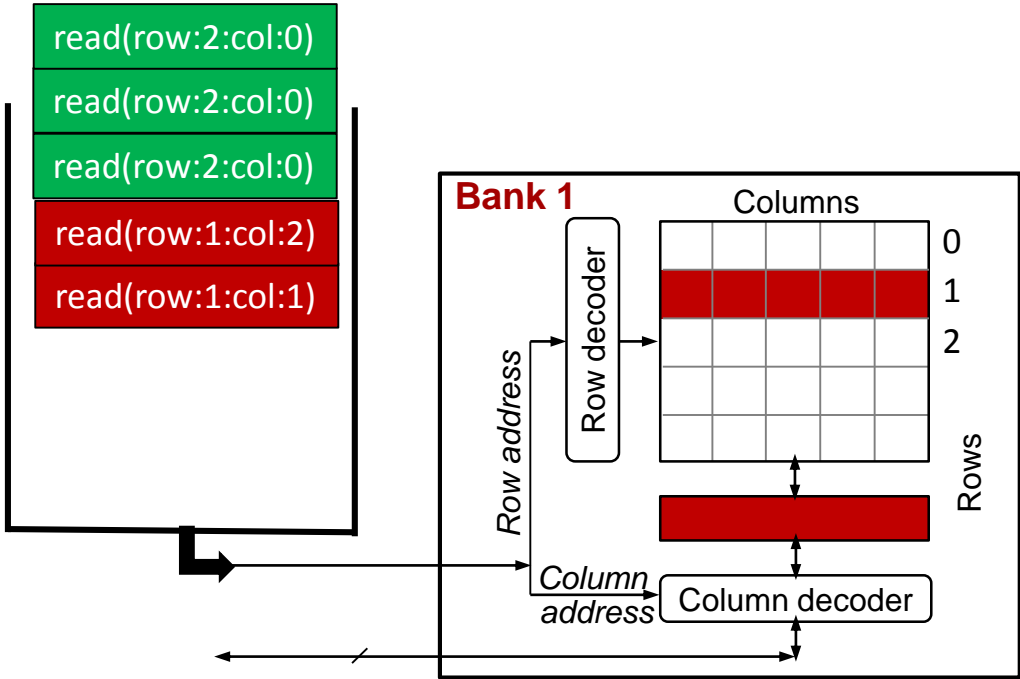
Memory Banks Delays



Memory Banks Delays



Memory Banks Delays



Other Shared Hardware

Direct Memory Access (DMA) I/O: Disk, Video cards, etc.

- Behave as another core that access memory

Table Lookaside Buffer (TLB)

- Cache of virtual-to-physical page for virtual memory
- Similar effect to shared cache but for virtual-to-physical memory translations