Bounding DRAM Interference in COTS Heterogeneous MPSoCs for Mixed Criticality Systems

Mohamed Hassan and Rodolfo Pellizzoni
Mixed Criticality Systems

MOTIVATION

• Emerging Systems No longer solely hosting isolated safety-critical tasks
  • Execute tasks with different criticalities

Bounding DRAM Interference in COTS Heterogeneous MPSoCs for Mixed Criticality Systems

(ABS)
• Engine Control Unit (ECU)
• Emerging Systems No longer solely hosting isolated safety-critical tasks
  • Execute tasks with different criticalities
  • Criticality $\alpha$ consequences of failure to meet requirements

High-criticality tasks
  • Airbag Control Unit (ACU)
  • Anti-lock Braking System (ABS)
  • Engine Control Unit (ECU)
• Emerging Systems No longer solely hosting isolated safety-critical tasks
  • Execute tasks with different criticalities
  • Criticality $\alpha$ consequences of failure to meet requirements

• Low-criticality tasks
  • Air Conditioning Unit
  • Connectivity Box
  • Infotainment Unit
Mixed Criticality Systems
Mixed Criticality Systems

MOTIVATION
Bounding DRAM Interference in COTS Heterogeneous MPSoCs for Mixed Criticality Systems
Why MPSoCs?

- Low cost
- High performance
- Energy Efficiency
- Low time-to-market (3rd party IPs)
Why Heterogenous MPSoCs?

- Variety of processing capabilities → Best-suits MCS conflicting requirements
Complementary SoC processor requirements

High performance compute
Infotainment
Cluster
Driver assist
Vehicle interface
User experience

Real-time control
Safe
Secure
Responsive
Reliable
Fast boot

Cost
Quality
Ecosystem
Temperature

Translating System-Level Requirements → SoC Level

Exploding Performance Requirements
- Rise of heterogeneous architectures & right-sized compute
- Cache coherency & End-to-end QoS of critical importance

Real-Time Sensor Processing
- Different IPs with differing requirements
- Ensuring communication happens without any deadlocks

Ultra-High Safety & Reliability
- Pressure to comply to industry standards – ISO 26262
- Functional Safety – Performance – Area Tradeoffs

Automotive Applications Require Different SoC Architectures

Need For Heterogeneous Computing

Image Acquisition
- Noise removal
- Pixel processing
- Image pyramids

Feature Extraction
- Optical flow
- Edge detection
- Gradient detection

Feature Processing
- Segmentation & filtering
- Object tracking
- Object detection

Pattern Recognition
- Feature reduction
- Feature classification
- Augmentation

Feedback and Action
- Computation & processing
- Feedback loop
- Avoidance signalling

- Smaller amounts of data
- Highly structured data
- Complex computation/item
- CPU
- DSP, Accel
- GPU, ISP
- Noise removal
- Segmentation & filtering
- Pixel processing
- Feature reduction
- Image pyramids
- Feature classification
- Edge detection
- Computation & processing
- Gradient detection
- Object detection
- Avoidance signalling
- Augmentation
Complementary SoC processor requirements

High performance compute
- Infotainment
- Cluster
- Driver assist
- Vehicle interface
- User experience

Cost  Quality  Ecosystem

Compute, Control, Sense

Real-time control

Automotive Applications Require Different SoC Architectures

Translating System-Level Requirement

Exploding Performance Requirements
- Rise of heterogeneous computing
- Cache colliions

Real-Time Sensor Processing
- Different I/O interfaces
- Ensuring real-time responsiveness

Ultra-High Safety & Reliability
- Pressure to comply to industry standards - ISO 26262
- Functional Safety - Performance - Area Tradeoffs

Safety and Security

Computation Automation
Sensing
Communication
Control Actuation

ARM® Cortex®-A
ARM® Cortex®-R
ARM® Cortex®-M

Computing
- Smaller amounts of data
- Highly structured data
- Complex computation/iterative

ARM

Feedback loop
- Noise removal
- Segmentation & filtering
- Image pyramids
- Edge detection
- Computation & processing
- Object tracking
- Avoidance signalling
- Augmentation
- Optical flow

Feature classification
- Feature recognition
- Augmentation
- Feedback loop
- Avoidance signalling
Heterogenous MPSoCs with Real-time Processors

MOTIVATION
Heterogenous MPSoCs with Real-time Processors
• DRAM Consists of multiple banks

Bounding DRAM Interference in COTS Heterogeneous MPSoCs for Mixed Criticality Systems

Background
• DRAM Consists of multiple banks
• DRAM Consists of multiple banks
• The memory controller (MC) manages accesses to DRAM
• DRAM Consists of multiple banks
• The memory controller (MC) manages accesses to DRAM
• A request in general consists of:
  • ACTIVATE command:
    • Bring data row from cells into sense amplifiers
• DRAM Consists of multiple banks
• The memory controller (MC) manages accesses to DRAM
• A request in general consists of:
  • ACTIVATE command:
    • Bring data row from cells into sense amplifiers
  • RD/WR commands:
    • To read/write from specific columns in the sense amplifiers
• DRAM Consists of multiple banks
• The memory controller (MC) manages accesses to DRAM
• A request in general consists of:
  • ACTIVATE command:
    • Bring data row from cells into sense amplifiers
  • RD/WR commands:
    • To read/write from specific columns in the sense amplifiers
  • PRECHARGE command:
    • to write back a previous row in the sense amplifiers before bringing the new one
• DRAM Consists of multiple banks
• The memory controller (MC) manages accesses to DRAM
• A request in general consists of:
  • ACTIVATE command:
    • Bring data row from cells into sense amplifiers
  • RD/WR commands:
    • To read/write from specific columns in the sense amplifiers
  • PRECHARGE command:
    • to write back a previous row in the sense amplifiers before bringing the new one
• All commands have associated timing constraints that have to be satisfied by the controller
System Overview

- **P processing elements**
  - $P_{cr}$ critical + $P_{ncr}$ non-critical
- **LLC** is write-back write-allocate
  - Writes to DRAM are only cache evictions
- **Single-channel single-rank DRAM subsystem**
- $N_B$ DRAM banks
System Overview

- **P processing elements**
  - $P_{cr}$ critical + $P_{ncr}$ non-critical
- LLC is write-back write-allocate
  - Writes to DRAM are only cache evictions
- Single-channel single-rank DRAM subsystem
- $N_B$ DRAM banks

Goal:
Derive an upper bound on the delay incurred by any memory request of a critical PE
System Details

Memory Behavior Depends on?:

- OS Configuration
- PE Architecture
- MC Policies

Shared cache(s)

Memory Controller

Off-chip Memory/ies

Applications

OS
System Details

Memory Behavior Depends on?:

- **Priority:**
  - PEs can be given priorities
  - COTS platforms support different priority levels
  - Existing analysis does not account for this

- **Intra-bank scheduling**
  - FR-FCFS
  - COTS also supports a threshold on reordering to prevent starvation

- **Inter-bank scheduling**
  - RR across banks
  - Two flavors:
    - Always schedule ready commands of any type (high performance)
    - Reorder only commands of different type (prevent starvation)

- **Read/Write arbitration, two flavors:**
  - Reads and writes have same priority
  - Serve in batches, where reads have higher priority
R/W Reorder
- 1: write batching
- 0: no write batching

FR-FCFS Threshold
- 1: FR-FCFS is capped
- 0: no cap on FR-FCFS

Priority
- 1: Critical PEs are higher priority
- 0: no priority

Inter-bank Reorder
- 1: Reorder across all commands
- 0: Reorder commands of different types

Pipeline
- IO-All: All PEs are In-order
- IO-Cr: Critical PEs are in-order
- OOO-All: All PEs are OOO

Partitioning
- No-Part: No Partitioning
- Part-Cr: Partition among critical apps
- Part-All: Partition among all apps
Platform Instances

R/W Reorder
• 1: write batching
• 0: no write batching

Inter-bank Reorder
• 1: Reorder across all commands
• 0: Reorder commands of diff types

FR-FCFS Threshold
• 1: FR-FCFS is capped
• 0: no cap on FR-FCFS

Priority
• 1: Critical PEs are higher priority
• 0: no priority

Pipeline
• IO-All: All PEs are In-order
• IO-Cr: Critical PEs are in-order
• OOO-All: All PEs are OOO

Partitioning
• No-Part: No Partitioning
• Part-Cr: Partition among critical apps
• Part-All: Partition among all apps

144 different platform instances!
## General Observations

### METHODOLOGY

144 different platform instances!

<table>
<thead>
<tr>
<th>OS</th>
<th>thr</th>
<th>pr</th>
<th>wb=0,breorder=0</th>
<th>wb=0,breorder=1</th>
<th>wb=1,breorder=0</th>
<th>wb=1,breorder=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part</td>
<td></td>
<td></td>
<td>OOO</td>
<td>IO-Cr</td>
<td>IO-All</td>
<td>OOO</td>
</tr>
<tr>
<td>Part-All</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No-Part</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part-Cr</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

144 Platform Instances
## General Observations

### METHODOLOGY

<table>
<thead>
<tr>
<th>OS</th>
<th>HW setup</th>
<th>part</th>
<th>thr</th>
<th>pr</th>
<th>wb=0, breorder=0</th>
<th>wb=0, breorder=1</th>
<th>wb=1, breorder=0</th>
<th>wb=1, breorder=1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>OOO IO-Cr IO-All</td>
<td>OOO IO-Cr IO-All</td>
<td>OOO IO-Cr IO-All</td>
<td>OOO IO-Cr IO-All</td>
</tr>
<tr>
<td>part</td>
<td>thr</td>
<td>pr</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part-All</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No-Part</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part-Cr</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Observation 1:** Unboundedness of inter-bank RR with reordering

If RR reorders across all commands (breorder=1) and no write batching is deployed (wb=0) → unbounded WCD
<table>
<thead>
<tr>
<th>OS</th>
<th>HW setup</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>wb=0,breorder=0</td>
<td>wb=0,breorder=1</td>
<td>wb=1,breorder=0</td>
<td>wb=1,breorder=1</td>
<td></td>
</tr>
<tr>
<td>part</td>
<td>OOO</td>
<td>IO-Cr</td>
<td>IO-All</td>
<td>OOO</td>
<td>IO-Cr</td>
</tr>
<tr>
<td>thr</td>
<td>pr</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 0</td>
<td>UNBOUNDED</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### General Observations

#### Observation 2:
- **Write batching effect**
- **Write batching cancels the effect of RR breorder:**
  - If $wb=1 \rightarrow breorder=x$

#### METHODOLOGY

<table>
<thead>
<tr>
<th>OS</th>
<th>thr</th>
<th>pr</th>
<th>wb=0,breorder=0</th>
<th>wb=0,breorder=1</th>
<th>wb=1,breorder=0</th>
<th>wb=1,breorder=1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>OOO</td>
<td>IO-Cr</td>
<td>IO-All</td>
<td>OOO</td>
</tr>
<tr>
<td><strong>part</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Part-All</strong></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>No-Part</strong></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Part-Cr</strong></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### General Observations

<table>
<thead>
<tr>
<th>OS</th>
<th>part</th>
<th>thr</th>
<th>pr</th>
<th>wb=0,breorder=0</th>
<th>wb=0,breorder=1</th>
<th>wb=1,breorder=0</th>
<th>wb=1,breorder=1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Part</td>
<td>0</td>
<td>0</td>
<td>OOO IO-Cr IO-All</td>
<td>OOO IO-Cr IO-All</td>
<td>OOO IO-Cr IO-All</td>
<td>OOO IO-Cr IO-All</td>
</tr>
<tr>
<td></td>
<td>All</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No-Part</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part-Cr</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**MMETHODOLOGY**

72
### General Observations

**Methodology**

<table>
<thead>
<tr>
<th>OS</th>
<th>HW setup</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>wb=0,breorder=0</td>
</tr>
<tr>
<td>part</td>
<td>thr</td>
</tr>
<tr>
<td>Part-All</td>
<td>0</td>
</tr>
<tr>
<td>Part-All</td>
<td>0</td>
</tr>
<tr>
<td>Part-All</td>
<td>1</td>
</tr>
<tr>
<td>Part-All</td>
<td>1</td>
</tr>
<tr>
<td>No-Part</td>
<td>0</td>
</tr>
<tr>
<td>No-Part</td>
<td>0</td>
</tr>
<tr>
<td>No-Part</td>
<td>1</td>
</tr>
<tr>
<td>No-Part</td>
<td>1</td>
</tr>
<tr>
<td>Part-Cr</td>
<td>0</td>
</tr>
<tr>
<td>Part-Cr</td>
<td>0</td>
</tr>
<tr>
<td>Part-Cr</td>
<td>1</td>
</tr>
<tr>
<td>Part-Cr</td>
<td>1</td>
</tr>
</tbody>
</table>

UNBOUNDED
### General Observations

**Observation 3:**
Unboundedness of FR-FCFS without threshold

If $\text{thr}=0 \land (\text{No-Part}) \lor ((\text{Part-Cr}) \land \text{pr}=0)$ \rightarrow Unbounded WCD

<table>
<thead>
<tr>
<th>OS</th>
<th>HW setup</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>wb=0,breorder=0</td>
</tr>
<tr>
<td>part</td>
<td>thr</td>
</tr>
<tr>
<td>Part-All</td>
<td>0 0</td>
</tr>
<tr>
<td></td>
<td>0 1</td>
</tr>
<tr>
<td></td>
<td>1 0</td>
</tr>
<tr>
<td></td>
<td>1 1</td>
</tr>
<tr>
<td>No-Part</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Part-Cr</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>OS</td>
<td>HW setup</td>
</tr>
<tr>
<td>----</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td><strong>thr</strong></td>
</tr>
<tr>
<td></td>
<td>OOO</td>
</tr>
<tr>
<td><strong>part</strong></td>
<td>0 0</td>
</tr>
<tr>
<td>Part-All</td>
<td>0 1</td>
</tr>
<tr>
<td></td>
<td>1 0</td>
</tr>
<tr>
<td></td>
<td>1 1</td>
</tr>
<tr>
<td><strong>No-Part</strong></td>
<td>0 0</td>
</tr>
<tr>
<td></td>
<td>0 1</td>
</tr>
<tr>
<td></td>
<td>1 0</td>
</tr>
<tr>
<td></td>
<td>1 1</td>
</tr>
<tr>
<td><strong>Part-Cr</strong></td>
<td>0 0</td>
</tr>
<tr>
<td></td>
<td>0 1</td>
</tr>
<tr>
<td></td>
<td>1 0</td>
</tr>
<tr>
<td></td>
<td>1 1</td>
</tr>
</tbody>
</table>

**General Observations**

**METHODOLOGY**
### General Observations

Table: HW setup

<table>
<thead>
<tr>
<th>OS</th>
<th>part</th>
<th>thr</th>
<th>pr</th>
<th>wb=0, breorder=0</th>
<th>wb=0, breorder=1</th>
<th>wb=1, breorder=x</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>OOO</td>
<td>IO-Cr</td>
<td>IO-All</td>
</tr>
<tr>
<td>Part-All</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part-Cr</td>
<td>0</td>
<td>0</td>
<td></td>
<td>UNBOUNDED</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Observation 4:**

Part-All effect

If Part-All → $r_{ua}$ does not suffer Intra-bank reordering or conflict interferences:

- $thr=x$
- If $wb=0$ → $pipe=x$

**Methodology**
## General Observations

### METHODOLOGY

<table>
<thead>
<tr>
<th>OS</th>
<th>HW setup</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>wb=0, breorder=0</td>
</tr>
<tr>
<td></td>
<td>OOO</td>
</tr>
<tr>
<td>part</td>
<td>thr</td>
</tr>
<tr>
<td>Part-All</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>No-Part</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Part-Cr</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

**Part**

- **UNBOUNDED**
- **UNBOUNDED**
- **UNBOUNDED**
- **UNBOUNDED**

**Thr**

- **wb=0, breorder=0**
- **wb=0, breorder=1**
- **wb=1, breorder=x**

**Pr**

- **config1**
- **config2**
- **config1**
- **config2**

**OOO**

- **config11**
- **config14**
- **config11**
- **config14**

**IO-Cr**

- **config12**
- **config15**
- **config12**
- **config15**

**IO-All**

- **config13**
- **config16**
- **config13**
- **config16**
### General Observations

**Observation 5:**

**Part-Cr effect when \( wb=0 \)**

If **Part-Cr & \( wb=0 \) →** \( r_{wb} \) does not suffer Intra-bank reordering nor conflict interferences from critical PEs:

- **IO-Cr** and **OOO-All** have same effect on WCD

---

**Table:**

<table>
<thead>
<tr>
<th>OS</th>
<th>HW setup</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>wb=0,breorder=0</td>
</tr>
<tr>
<td></td>
<td>wb=0,breorder=1</td>
</tr>
<tr>
<td></td>
<td>wb=1, breorder=x</td>
</tr>
<tr>
<td></td>
<td>OOO</td>
</tr>
<tr>
<td></td>
<td>OOO</td>
</tr>
<tr>
<td></td>
<td>OOO</td>
</tr>
</tbody>
</table>

- **part**
  - **thr**
  - **pr**
  - **config1**
  - **config2**
  - **config14**
  - **config15**
  - **config16**
  - **config11**
  - **config12**
  - **config13**
  - **config**
  - **UNBOUND**
  - **UNBOUND**
## General Observations

### METHODOLOGY

<table>
<thead>
<tr>
<th>Part-Cr</th>
<th>No-Part</th>
<th>Part-All</th>
<th>OS</th>
<th>thr</th>
<th>pr</th>
<th>wb-0, breorder=0</th>
<th>wb-0, breorder=1</th>
<th>wb-1, breorder=x</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>UNBOUNDED</td>
<td>UNBOUNDED</td>
<td>UNBOUNDED</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>UNBOUNDED</td>
<td>UNBOUNDED</td>
<td>UNBOUNDED</td>
</tr>
</tbody>
</table>

*[Diagram and table content]*
### General Observations

#### Observation 6:
**Priority effect when \( wb=0 \)**

If \( pr=1 \) & \( wb=0 \) → pipeline architecture of non-critical PEs has no effect on WCD:

- IO-Cr and IO-All have same effect on WCD

---

<table>
<thead>
<tr>
<th>OS</th>
<th>HW setup</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( wb=0, breorder=0 )</td>
</tr>
<tr>
<td>part</td>
<td>thr</td>
</tr>
<tr>
<td>Part-All</td>
<td></td>
</tr>
<tr>
<td>0 0</td>
<td>config1</td>
</tr>
<tr>
<td>0 1</td>
<td>config2</td>
</tr>
<tr>
<td>1 0</td>
<td>config14</td>
</tr>
<tr>
<td>1</td>
<td>config11</td>
</tr>
<tr>
<td>No-Part</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>config8</td>
</tr>
<tr>
<td>1</td>
<td>config9</td>
</tr>
<tr>
<td>Part-Cr</td>
<td></td>
</tr>
<tr>
<td>0 0</td>
<td>UNBOUNDED</td>
</tr>
<tr>
<td>0 1</td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>config10</td>
</tr>
</tbody>
</table>

---

*METHODOLOGY*
## General Observations

### Methodology

<table>
<thead>
<tr>
<th>OS</th>
<th>HW setup</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>wb=0,</td>
<td>wb=0,</td>
<td>wb=1,</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>breorder=0</td>
<td>breorder=0</td>
<td>breorder=x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>OOO</td>
<td>IO-Cr</td>
<td>IO-All</td>
<td>OOO</td>
<td>IO-Cr</td>
<td>IO-All</td>
<td>OOO</td>
<td>IO-Cr</td>
<td>IO-All</td>
<td></td>
</tr>
<tr>
<td>part</td>
<td>thr</td>
<td>pr</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-----</td>
<td>----</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>Part-All</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>config1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No-Part</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UNBOUND</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Config7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part-Cr</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UNBOUND</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### General Observations

<table>
<thead>
<tr>
<th>OS</th>
<th>hw setup</th>
<th>000</th>
<th>IO-Cr</th>
<th>IO-All</th>
</tr>
</thead>
<tbody>
<tr>
<td>part</td>
<td>thr</td>
<td>pr</td>
<td>wb=0, breorder=0</td>
<td>wb=0, breorder=1</td>
</tr>
<tr>
<td>Part-All</td>
<td>0</td>
<td>0</td>
<td>config1</td>
<td>config11, config12, config13</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>config2</td>
<td>config14, config15, config16</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>config1</td>
<td>config11, config12, config13</td>
</tr>
<tr>
<td>No-Part</td>
<td>0</td>
<td>0</td>
<td>config1</td>
<td>config1</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>config1</td>
<td>config1</td>
</tr>
<tr>
<td>Part-Cr</td>
<td>0</td>
<td>0</td>
<td>config8</td>
<td>UNBOUNDED</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>config9</td>
<td>UNBOUNDED</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>config10</td>
<td>UNBOUNDED</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>config10</td>
<td>UNBOUNDED</td>
</tr>
</tbody>
</table>

**Observation 7:**

**Priority with Part-Cr effect**

- \( \text{thr} = x \)
- If \( \text{wb} = 0 \) → \( \text{pipe} = x \)

*Same as Part-All effect!!*
### General Observations

<table>
<thead>
<tr>
<th>OS</th>
<th>HW setup</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>wb=0, breorder=0</td>
</tr>
<tr>
<td></td>
<td>OOO</td>
</tr>
<tr>
<td>part</td>
<td>thr</td>
</tr>
<tr>
<td>Part:All</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>No:Part</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Part:Cr</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

#### METHODOLOGY
### General Observations

144 different platform instances!

#### METHODOLOGY

<table>
<thead>
<tr>
<th>OS</th>
<th>HW setup</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>wb=0, breorder=0</td>
</tr>
<tr>
<td>part</td>
<td>thr</td>
</tr>
<tr>
<td>Part:All</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>No-Part</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Part:Cr</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

144 Instances $\rightarrow$ 28 Configurations
• Consider all timing constraints generated by commands of interfering requests of other PEs serviced between the times when $r_{ua}$ arrives and finishes

• + Delays due to command bus contention

• Compute WCD for each configuration?
• Consider all timing constraints generated by commands of interfering requests of other PEs serviced between the times when $r_{ua}$ arrives and finishes

• + Delays due to command bus contention

• **Compute WCD for each configuration?**
  • Still too much
  • Not general enough

---

**Memory Delay Building Blocks**
• Consider all timing constraints generated by commands of interfering requests of other PEs serviced between the times when $r_{ua}$ arrives and finishes

• + Delays due to command bus contention

• Compute WCD for each configuration?
  • Still too much
  • Not general enough

• Configuration-independent DRAM delay components

---

Memory Delay Building Blocks
• We classify interfering requests (aka delay sources) into four types → causing four basic interferences:
• We classify interfering requests (aka delay sources) into four types → causing four basic interferences:
  1. Inter-bank interference (requests to other banks)

\[
WCD = L_{\text{InterB}}(N_{\text{InterB}}, wb)
\]
We classify interfering requests (aka delay sources) into four types → causing four basic interferences:

1. Inter-bank interference (requests to other banks)
2. Write batch Interference (only for R/W reordering)

\[ WCD = L_{\text{Inter}B}(N_{\text{Inter}B}, wb) + wb \times L^{WB}(N^{WB}) \]
We classify interfering requests (aka delay sources) into four types → causing four basic interferences:

1. Inter-bank interference (requests to other banks)
2. Write batch Interference (only for R/W reordering)
3. Conflict interference (requests to same bank different rows arrived before $r_{ua}$)

$$WCD = L_{InterB}(N_{InterB}, wb) + wb \times L_{WB}(N_{WB}) + L_{Conf}(N_{Conf})$$
• We classify interfering requests (aka delay sources) into four types → causing four basic interferences:
  1. Inter-bank interference (requests to other banks)
  2. Write batch Interference (only for R/W reordering)
  3. Conflict interference (requests to same bank different rows arrived before \( r_{ua} \))

\[
WCD = L^{\text{InterB}}(N^{\text{InterB}}, wb) + wb \times L^{\text{WB}}(N^{\text{WB}}) + L^{\text{Conf}}(N^{\text{Conf}}) + N^{\text{Conf}} \times L^{\text{InterB}}(N^{\text{InterB}}, wb)
\]
• We classify interfering requests (aka delay sources) into four types → causing four basic interferences:
  1. Inter-bank interference (requests to other banks)
  2. Write batch Interference (only for R/W reordering)
  3. Conflict interference (requests to same bank different rows arrived before $r_{ua}$)
  4. Intra-bank Reorder interference (FR-FCFS)

\[
WCD = L^{\text{InterB}}(N^{\text{InterB}}, wb) \\
+ wb \times L^{\text{WB}}(N^{\text{WB}}) \\
+ L^{\text{Conf}}(N^{\text{Conf}}) \\
+ N^{\text{Conf}} \times L^{\text{InterB}}(N^{\text{InterB}}, wb) \\
+ L^{\text{Reorder}}(N^{\text{Reorder}}, wb)
\]
We classify interfering requests (aka delay sources) into four types → causing four basic interferences:

1. Inter-bank interference (requests to other banks)
2. Write batch Interference (only for R/W reordering)
3. Conflict interference (requests to same bank different rows arrived before \( r_{ua} \))
4. Intra-bank Reorder interference (FR-FCFS)

\[
WCD = L^{\text{InterB}}(N^{\text{InterB}}, wb) + wb \times L^{\text{WB}}(N^{\text{WB}}) + L^{\text{Conf}}(N^{\text{Conf}}) + N^{\text{Conf}} \times L^{\text{InterB}}(N^{\text{InterB}}, wb) + L^{\text{Reorder}}(N^{\text{Reorder}}, wb) + N^{\text{Reorder}} \times L^{\text{CAS}}(N^{\text{InterB}}, wb)
\]
• Let’s assume we know # of interfering requests ($Ns$), how to compute the latency components ($Ls$)?

$\rightarrow Ls$ only depend on $Ns$ and JEDEC “known” timing constraints.

$$WCD = L_{\text{InterB}}(N_{\text{InterB}}, wb) + \text{wb} \times L_{\text{WB}}(N_{\text{WB}}) + L_{\text{Conf}}(N_{\text{Conf}}) + N_{\text{Conf}} \times L_{\text{InterB}}(N_{\text{InterB}}, \text{wb}) + L_{\text{Reorder}}(N_{\text{Reorder}}, \text{wb}) + N_{\text{Reorder}} \times L_{\text{CAS}}(N_{\text{InterB}}, \text{wb})$$
• Let’s assume we know # of interfering requests (Ns), how to compute the latency components (Ls)?
  → Ls only depend on Ns and JEDEC “known” timing constraints
  → $L^{Conf}$ as example

$$L^{Conf}(N^{Conf}) = N^{Conf} \times (\text{MAX}(t_{RAS}, t_{RCD} + t_{WL} + t_{B} + t_{WR}) + t_{RP})$$
• Let’s assume we know # of interfering requests (Ns), how to compute the latency components (Ls)?

→ Ls only depend on Ns and JEDEC “known” timing constraints
→ $L^{Conf}$ as example

$L^{Conf} (N^{Conf}) = N^{Conf} \times (MAX(t_{RAS}, t_{RCD} + t_{WL} + t_B + t_{WR}) + t_{RP})$

• Configuration-independent DRAM delay components 😊
• Let’s assume we know # of interfering requests (Ns), how to compute the latency components (Ls)?
  → Ls only depend on Ns and JEDEC “known” timing constraints

• Now: It only remains to compute the Ns.

• Configuration-independent DRAM delay components 😊
Now: It only remains to compute the $N_s$. $\rightarrow$ Config. dependent
Now: It only remains to compute the $N_s$. → Config. dependent

Take config3 as an example:

config3:
- no WB
- FR-FCFS thr
- no FP
- Inter-bank reorder among different types only (breorder=0)
- All PEs are OOO
- no partitioning
• Now: It only remains to compute the $N_s$. $\rightarrow$ Config. dependent
• Take config3 as an example:

1. **Conflicts ($N^{Conf}$):**
   - OOO-All $\rightarrow$ each PE has PR pending reqs
   - No FP $\rightarrow$ critical and non-critical scheduled similarly
   - Then $N^{Conf} = (P - 1) \times PR$ requests can conflict with $r_{ua}$

**config3:**
- no WB
- FR-FCFS thr
- no FP
- Inter-bank reorder among different types only ($breorder=0$)
- All PEs are OOO
- no partitioning
• Now: It only remains to compute the $N$s. → Config. dependent
• Take config3 as an example:

1. **Conflicts ($N^C$):**
   - OOO-All → each PE has PR pending reqs
   - No FP → critical and non-critical scheduled similarly
   - Then $N^C = (P - 1) \times PR$ requests can conflict with $r_{ua}$

2. **Reorder ($N^R$):**
   - FR-FCFS thr → max of $N^R = N_{thr}$ requests can be reordered before $r_{ua}$

**config3:**
- no WB
- FR-FCFS thr
- no FP
- Inter-bank reorder among different types only (breorder=0)
- All PEs are OOO
- no partitioning
Now: It only remains to compute the $N$s. → Config. dependent
Take config3 as an example:

1. **Conflicts ($N^{Conf}$):**
   - OOO-All → each PE has PR pending reqs
   - No FP → critical and non-critical scheduled similarly
   - Then $N^{Conf} = (P - 1) \times PR$ requests can conflict with $r_{ua}$

2. **Reorder ($N^{Reorder}$):**
   - FR-FCFS thr → max of $N^{Reorder} = N_{thr}$ requests can be reordered before $r_{ua}$

3. **Inter-bank ($N^{InterB}$):**
   - RR arbiter and no FP → max of $N^{InterB} = N_{B} - 1$ reqs from other banks can be reordered before $r_{ua}$

config3:
- no WB
- FR-FCFS thr
- no FP
- Inter-bank reorder among different types only (breorder=0)
- All PEs are OOO
- no partitioning
Now: It only remains to compute the Ns. \( \rightarrow \) Config. dependent
Take config3 as an example:

1. **Conflicts** \( N^{Conf} \):
   - OOO-All \( \rightarrow \) each PE has PR pending reqs
   - No FP \( \rightarrow \) critical and non-critical scheduled similarly
   - Then \( N^{Conf} = (P - 1) \times PR \) requests can conflict with \( r_{ua} \)

2. **Reorder** \( N^{Reorder} \):
   - FR-FCFS thr \( \rightarrow \) max of \( N^{Reorder} = N_{thr} \) requests can be reordered before \( r_{ua} \)

3. **Inter-bank** \( N^{InterB} \):
   - RR arbiter and no FP \( \rightarrow \) max of \( N^{InterB} = N_{B} - 1 \) reqs from other banks can be reordered before \( r_{ua} \)

4. **Write Batch** \( N^{WB} \):
   - No WB \( \rightarrow N^{WB} = 0 \)

**config3:**
- no WB
- FR-FCFS thr
- no FP
- Inter-bank reorder among different types only (breorder=0)
- All PEs are OOO
- no partitioning

# of Interfering Requests
• Follow Same approach for all configurations

1. Conflicts ($N_{Conf}$):
   - OOO-All → each PE has PR pending reqs
   - No FP → critical and non-critical scheduled similarly
   - Then $N_{Conf} = (P - 1) \cdot PR
2. Reorder ($N_{Reorder}$):
   - FR-FCFS thr → max of $N_{Reorder} = N_{thr}$ requests can be reordered before $r_{ua}$
3. Inter-bank ($N_{InterB}$):
   - RR arbiter and no FP → max of $N_{InterB} = N_{B} - 1$ reqs from other banks can be reordered before $r_{ua}$
4. Write Batch ($N_{WB}$):
   - No WB → $N_{WB} = 0$
| PEs | • A private 16KB L1 and a shared 1MB L2 cache  
• An in-order PE has a maximum of one pending request to the DRAM  
• An OOO PE has a maximum of 4 pending requests to the DRAM (PR = 4)  
• Four-processor system unless otherwise specified |
|-----|--------------------------------------------------|
| OS Mapping | • Through the virtual-to-physical address mapping component at MacSim’s frontend  
• Based on the configuration, we enable the corresponding partitioning (Part-All, Part-Cr, or No-Part) |
| DRAM | DDR3-1333H with single channel, single rank, and 8 banks |
| MC | • Based on the configuration,  
• Per-bank queues with RR among banks and FR-FCFS arbitration within each bank  
• Based on the configuration:  
  • critical PEs can be assigned higher priority than non-critical PEs  
  • enable or disable the threshold for FR-FCFS  
  • For enabled threshold: \( N_{thr} = 8 \), unless otherwise specified  
  • enable or disable write batching |
| Benchmarks | EEMBC Automotive  
• The two critical PEs execute a2time and rspeed  
• The two non-critical PEs execute matrix and aifftr |
| | Synthetic  
• Each of the critical PEs execute one instance of the latency benchmark  
• Each of the non-critical PEs execute one instance of the Bandwidth benchmark |
WCD of Critical Processors

RESULTS

<table>
<thead>
<tr>
<th>config</th>
<th>IO-All</th>
<th>IO-Cr</th>
<th>OOO-All</th>
</tr>
</thead>
<tbody>
<tr>
<td>noPr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>noThr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>thr</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>config</th>
<th>IO-All</th>
<th>IO-Cr</th>
<th>OOO-All</th>
</tr>
</thead>
<tbody>
<tr>
<td>pr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>noThr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>thr</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>config</th>
<th>IO-All</th>
<th>IO-Cr</th>
<th>OOO-All</th>
</tr>
</thead>
<tbody>
<tr>
<td>pr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>noThr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>thr</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>config</th>
<th>IO-All</th>
<th>IO-Cr</th>
<th>OOO-All</th>
</tr>
</thead>
<tbody>
<tr>
<td>pr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>noThr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>thr</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. Intra-bank reorder (FRFCFS) has no effect on WCD

WCD of Critical Processors

No bank sharing (Part-All)
Captures the interrelation amongst memory requests. 1. Intra-bank reorder (FRFCFS) has no effect on WCD.

For same priority setting: pipeline has no effect on WCD.

No bank sharing (*Part-All*)

1. Intra-bank reorder (FRFCFS) has no effect on WCD.

2. For same priority setting: pipeline has no effect on WCD.
1. Intra-bank reorder (FRFCFS) has no effect on WCD.

No bank sharing (Part-All)

2. For same priority setting: pipeline has no effect on WCD.

3. Part-All + FP gives lowest WCD across all configurations.

WCD of Critical Processors
4. With priority: the pipeline of non-critical processors is irrelevant.

WCD of Critical Processors
4. With priority: the pipeline of non-critical processors is irrelevant

5. No-Part + No-FP gives highest WCD across all configurations
Part-Cr + FP significantly reduces WCD.
Compared to Config 6 (No-Part):
- Config 2 (Part-All):
  - 96% less WCD
  - 60% BW degradation
- Config 8 (Part-Cr + FP):
  - 89% less WCD
  - 0.85% BW degradation
• Normalized to WB-expr
• WB-analytical is very pessimistic
• WB improves avg case
  • noWb-expr is 2.84x on average as compared to Wb-expr
  • even reaches 10x

Write Batching Effect
• Config. 1 & 2 & 8 offers complete isolation
• Config. 1 and 2: *Part-All*

• Config. 8: *Part-Cr with fixed priority*

**Sensitivity to # Processors**

**RESULTS**
• Config 1 & 2 & 8 offers complete isolation
• Config 6 & 7 offers isolation from non-cr PEs

Sensitivity to # Processors

RESULTS
- Config 1 & 2 & 8 offers complete isolation
- Config 6 & 7 offers isolation from non-cr PEs
- Config 3,4,9 are more vulnerable to WCD↑ when Pncr↑

**RESULTS**

- Config. 6 & 7: deploy fixed priority
- Config. 3&4&9: non-cr PEs are OOO
- Config. 5&10: non-cr PEs are in-order
• Config 1 & 2 & 8 offers complete isolation
• Config 6 & 7 offers isolation from non-cr PEs
• Config 3, 4, 9 are more vulnerable to WCD↑ when Pncr↑

In fact, slope is 4x, why -> #pending reqs is 4
• Config 1 & 2 & 8 offers complete isolation
• Config 6 & 7 offers isolation from non-cr PEs
• Config 9 & 10 offers isolation from cr PEs
• Config 3, 4, 9 are more vulnerable to WCD when Pncr
• Config 3, 6 are more vulnerable to WCD when Pcr

Sensitivity to # Processors
Sensitivity to FR-FCFS thr.

RESULTS

![Graph showing sensitivity to FR-FCFS thr.](image-url)
• Heterogeneous MPSoCs are important for Mixed Criticality Systems
• Heterogeneous MPSoCs are important for Mixed Criticality Systems
• We derived a generalized analysis that bounds the per-request DRAM interference delay in MPSoCs
• Heterogeneous MPSoCs are important for Mixed Criticality Systems
• We derived a generalized analysis that bounds the per-request DRAM interference delay in MPSoCs
• We derived a generalized analysis that bounds the per-request DRAM interference delay in MPSoCs
We derived a generalized analysis that bounds the per-request DRAM interference delay in MPSoCs.
We derived a generalized analysis that bounds the per-request DRAM interference delay in MPSoCs.
Summary & Conclusions

- We derived a generalized analysis that bounds the per-request DRAM interference delay in MPSoCs.

**General Observations**

- **Inter-bank Reorder**
  - 1: Reorder across all commands
  - 0: Reorder commands of different types

- **Priority**
  - 1: Critical PEs are higher priority
  - 0: No priority

- **Inter-platform Reorder**
  - 1: All PEs are in order
  - 0: All PEs are unordered

- **Pipeline**
  - 1: FR-FCFS is capped
  - 0: No cap on FR-FCFS

- **R/W Reorder**
  - 1: Write batching
  - 0: No write batching

- **Partitioning**
  - 1: Part among critical apps
  - 0: No partitioning

- **IO**
  - 1: All PEs are in order
  - 0: All PEs are unordered

- **OOO**
  - 1: All PEs are OOO

- **MC Policies**
  - 1: Off-chip
  - 0: Shared Memory

- **Memory Behavior Depends on?**
  - OS Configuration
  - PE Architecture
  - MC Policies

144 different platform instances! 28 configurations

Compute the number of interfering requests

Applications

OS

OS Configuration

PE Architecture

MC Policies

144 instances

144 different platform instances!
• We derived a generalized analysis that bounds the per-request DRAM interference delay in MPSoCs

144 different platform instances!

Inter-bank Recorder
- 1: Recorder across all banks
- 0: Recorder only in bank under analysis

FR-FCFS Threshold
- 1: FR-FCFS is capped
- 0: no cap on FR-FCFS

Priority
- 1: Critical PEs are higher priority
- 0: no priority

IO
- All: All PEs are In-order
- Cr: Critical PEs are In-order
- OOO: All PEs are OOO

Partitioning
- No-Part: No Partitioning
- Part-Cr: Partition among critical apps
- Part-All: Partition among all apps

Intra-bank Reorder
- Requests serviced before the one under analysis because they are ready, i.e., targeting the open row in the bank

Inter-bank Reorder
- Requests serviced before the one under analysis because they are ready, i.e., targeting the open row in the bank

Write Batching
- Delay incurred by a read waiting for a write batch to finish
- Applies only for configurations with write batching

Compute the number of interfering requests

Worst Case Delay (WCD)
• We derived a generalized analysis that bounds the per-request DRAM interference delay in MPSoCs.

Main lessons:
1. DRAM’s WCD significantly depends on MPSoC features.
Main lessons:
1. DRAM’s WCD significantly depends on MPSoC features
2. Identified features that lead to unbounded WCD
Main lessons:

1. DRAM’s WCD significantly depends on MPSoC features
2. Identified features that lead to unbounded WCD
3. Leveraging existing features such as PE prioritization can allow the designer to better trade-off the maximum delay for critical applications and the bandwidth for non-critical ones.

Summary & Conclusions
Summary & Conclusions

Main lessons:
1. DRAM’s WCD significantly depends on MPSoC features
2. Identified features that lead to unbounded WCD
3. Leveraging existing features such as PE prioritization can allow the designer to better trade-off the maximum delay for critical applications and the bandwidth for non-critical ones.
4. There is interdependency among the effects of the features on both the delay and the bandwidth. Existence of some features can countermand the effect of other features.

N/W Reorder
- 1: write batching
- 0: no write batching

FR-FCFS Threshold
- 1: FR-FCFS is capped
- 0: no cap on FR-FCFS

Priority
- 1: Critical PEs are higher priority
- 0: no priority

Inter-bank Reorder
- 1: Reorder across all commands
- 0: Reorder commands of different types

Pipeline
- IO-All: All PEs are in-order
- IO-Cr: Critical PEs are in-order
- OOO-All: All PEs are OOO

Partitioning
- No-Part: No Partitioning
- Part-Cr: Partition among critical apps
- Part-All: Partition among all apps

94 different platform instances! 28 configurations

Row Conflict
Requests arrived before the one under analysis and are targeting different rows

Intra-bank Reorder
Requests serviced before the one under analysis because they are ready, i.e., targeting the open row in the bank

Write Batching
Delay incurred by a read waiting for a write batch to finish
Applies only for configurations with write batching

Delay Basic Building Blocks
N WB
N InterB Conf

Worst Case Delay (WCD)
Compute the number of interfering requests N Conf, N Reorder, N InterB, N WB

General Observations
PE 2
PE 4
PE 1
PE 3
Shared latency
All commands
of Shared
Memory
Applications
OS
Memory
Dependence
MC Policies
PE Architecture
OS Configuration
Depends on?
Main lessons:
1. DRAM’s WCD significantly depends on MPSoC features
2. Identified features that lead to unbounded WCD
3. Leveraging existing features such as PE prioritization can allow the designer to better trade-off the maximum delay for critical applications and the bandwidth for non-critical ones.
4. There is interdependency among the effects of the features on both the delay and the bandwidth. Existence of some features can countermand the effect of other features.
Main lessons:

1. DRAM’s WCD significantly depends on MPSoC features
2. Identified features that lead to unbounded WCD
3. Leveraging existing features such as PE prioritization can allow the designer to better trade-off the maximum delay for critical applications and the bandwidth for non-critical ones.
4. There is interdependency among the effects of the features on both the delay and the bandwidth. Existence of some features can countermand the effect of other features
5. Although write batching mechanism works well in the average case, it unfortunately induces pathological cases that result in high bounds on per-request delay

Summary & Conclusions
Main lessons:
1. DRAM’s WCD significantly depends on MPSoC features
2. Identified features that lead to unbounded WCD
3. leveraging existing features such as PE prioritization can allow the designer to better trade-off the maximum delay for critical applications and the bandwidth for non-critical ones.
4. There is interdependency among the effects of the features on both the delay and the bandwidth. Existence of some features can countermand the effect of other features
5. Although write batching mechanism works well in the average case, it unfortunately induces pathological cases that result in high bounds on per-request delay
- Config 1 & 2 & 8 offers complete isolation from FR-FCFS reordering

- Config. 1 and 2: Part-All

- Config. 8: Part-Cr with fixed priority
• Config 1 & 2 & 8 offers complete isolation from FR-FCFS threshold
• Configs 3-7 & 10 scales linearly with FR-FCFS threshold

WCD

thr

• Slope is the same for these configs
• $L^{Reorder}$ component depends only on thr and JEDEC constraints
• Reordering has huge impact on WCD