

parMERASA

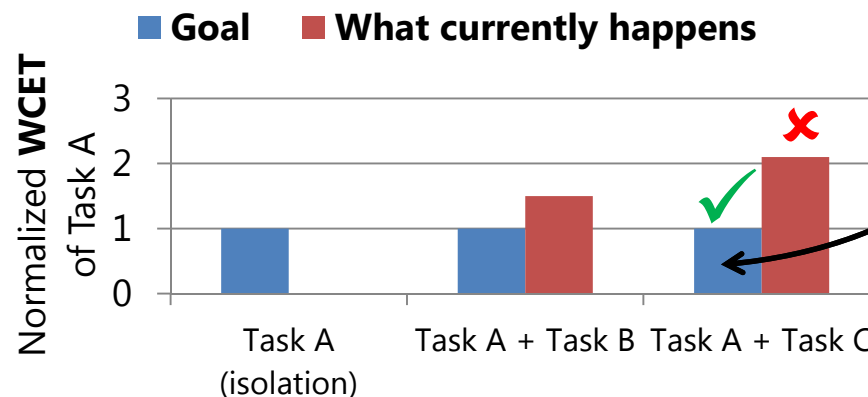
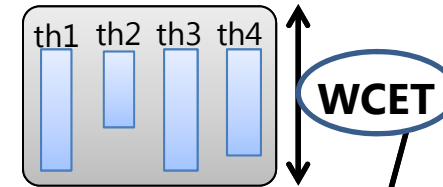
Many-core Processor Architecture

Dr. Eduardo Quiñones

Dr. Sascha Uhrig



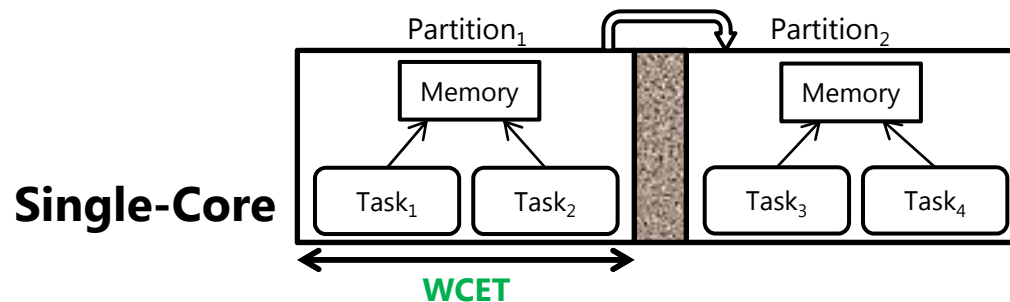
- **Parallel execution** of hard real-time applications
 - Hardware support for WCET analysis
- **Mixed-critical** applications, i.e. simultaneous execution of hard and none real-time applications without interferences
 - Timing isolation must be guaranteed (time composability)



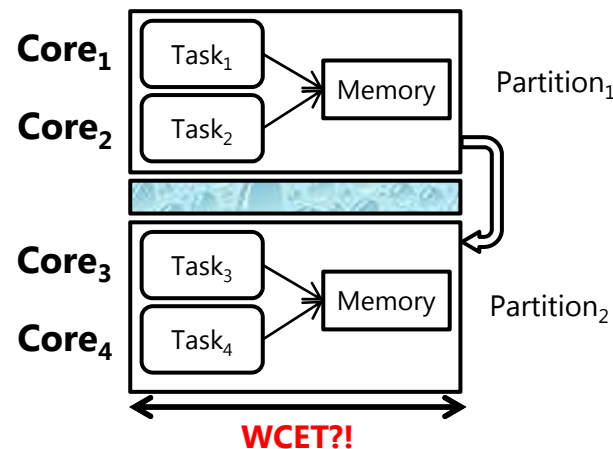
- Current safety critical real-time systems in single-core execution
 - Software Partitions
- Providing **timing isolation** properties to many-core architectures
 - Parallel Software Partitions (software)
 - Guaranteed Resource Partitions (hardware)
- Providing **time-aware hardware** mechanisms
 - Design of Network on Chip
 - Predictable Cache Coherence Mechanisms
- parMERASA focuses on providing time guarantees
 - Functional isolation are considered although not the main focus

- Safety critical real-time systems rely on **incremental qualification** (avionics) or **freedom from interference** (automotive)
 - Allows applications to be subject to formal certification in isolation and independently of other applications
 - Enabled by guaranteeing **robust space and time partitioning** among applications
- The ARINC 653 (avionics) and the ISO 26262 (automotive) standards define the **software partitions** as the incarnation of the robust partitioning
 - Applications are encapsulated within software partitions
 - A mechanism to prevent interferences between applications in terms of communication and execution time

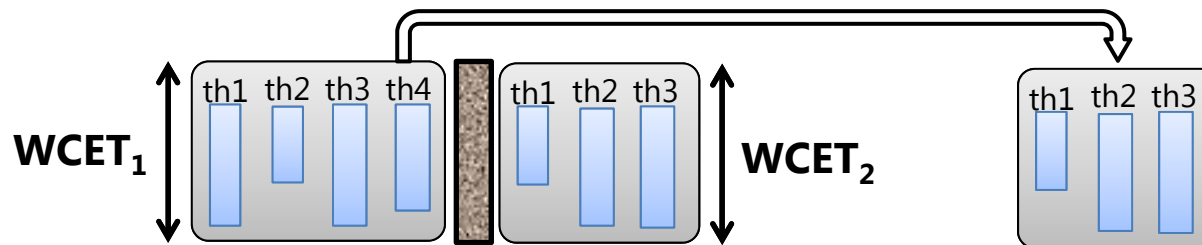
- Communication standards are defined two isolation timing properties
 - **Intra-partition** communication to exchange data among tasks (threads) that form an application through global variables
 - **Inter-partition** communication to exchange data among partitions through messages and define run-after dependencies
 - The destination partition is not executed
 - The inter-partition communication is known at integration time
- In single-core, timing properties are guaranteed by defining a dedicated **time window** per each partition



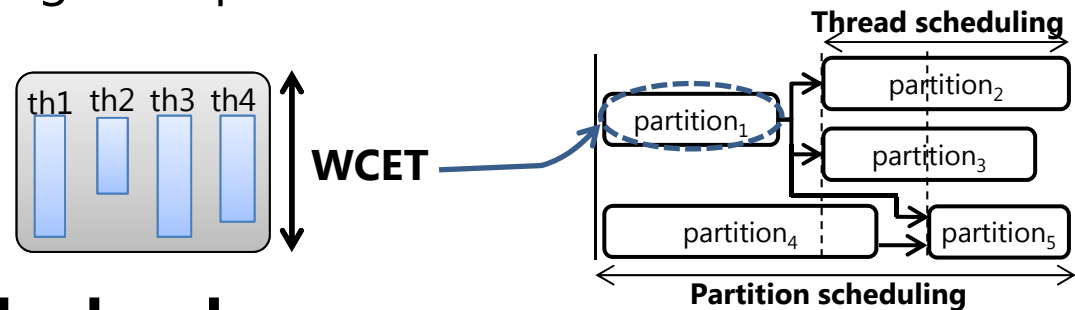
- Timing properties of software partitions are not guarantee anymore
 - The access to processor resources are **not exclusive**
 - The intra-partition communication from tasks belonging to different partitions may **interference** among them
 - The inter-partition communication may **affect the execution time** of other partitions being run.



- Robust time partitioning property
 - Parallel threads (tasks) of one software partition **cannot affect** the behavior of parallel threads (tasks) of other software partitions being run simultaneously
 - Intra-partition communication cannot exceed the partition
 - The impact of **inter-partition** communication must be considered at integration time
 - The destination partition cannot be executed in parallel
 - Parallel threads (tasks) of the same software partition **do not require** to fulfill the robust partitioning
 - Timing analysis is provided at application level



- Allows threads and partitions to be allocated in two independent phases
 - Thread phase**, parallel threads are mapped to a set of processor resources
 - Partition phase**, partitions are allocated to the many-core processor considering inter-partition communication

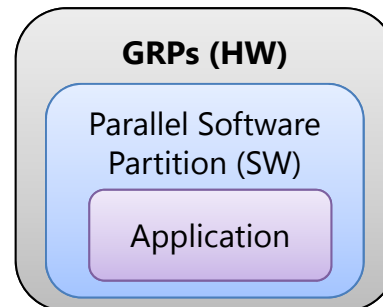


- But, what about the hardware...**

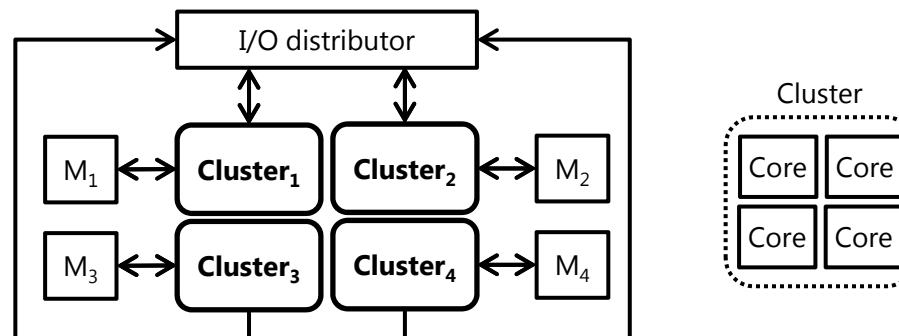


■ Guaranteed Resource Partition (GRP)

- Hardware execution environment composed of a **pool of processor resources** (including cores, NoC, memory devices, etc.) that guarantees partitioning in which parallel software partitions run
 - Intra-partition communication requests cannot exceed GRPs
 - Inter-partition communication requests must be considered at integration time
- GRPs are the hardware counterpart of parallel software partitions
 - Parallel software partitions are encapsulated within a GRP

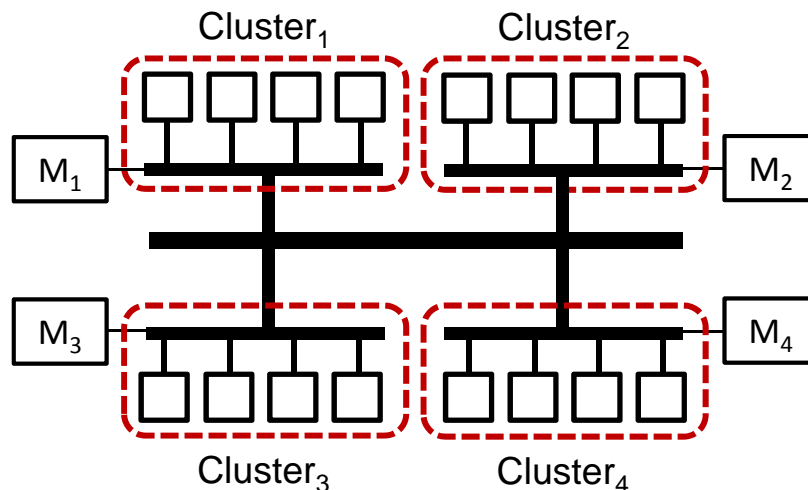


- To support GRPs, parMERASA proposes clusterized architectures
 - Allow cores to be organized in **(virtual) clusters**
 - A **NoC** connecting cores and clusters
 - A **private memory resource** per cluster
- **GRP → Cluster**

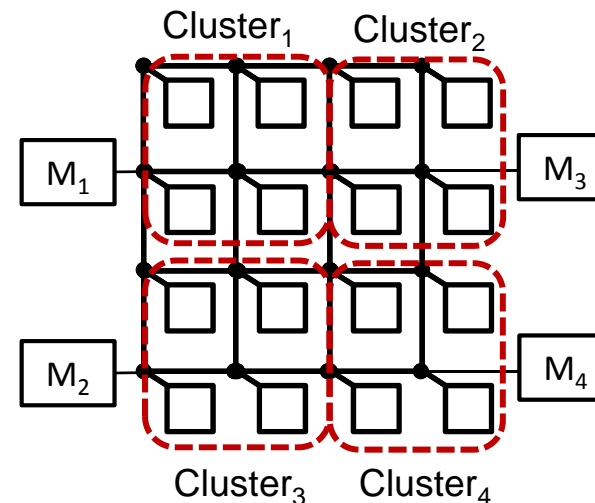


- Defines a set of cores assigned to a GRP in which the software partition executes
 - **Intra-partition** communication requests **cannot exceed** cluster boundaries

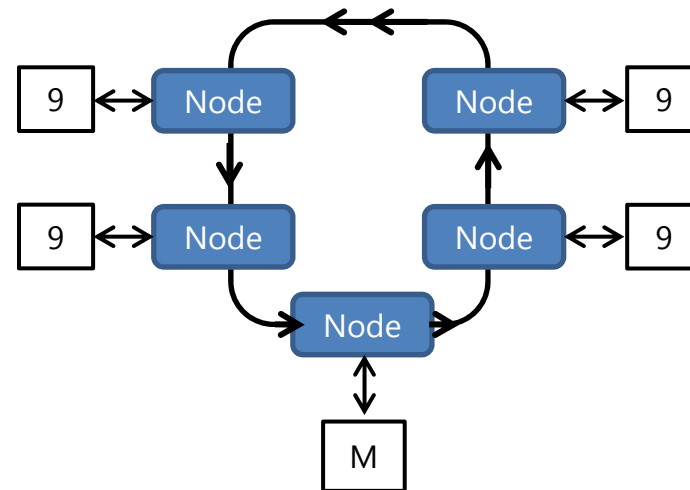
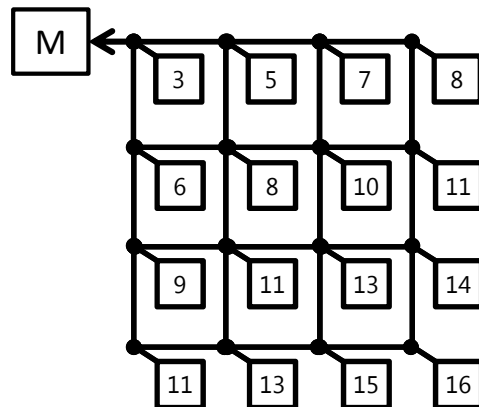
Hardware Cluster Definition



Virtual Cluster Definition XY routing

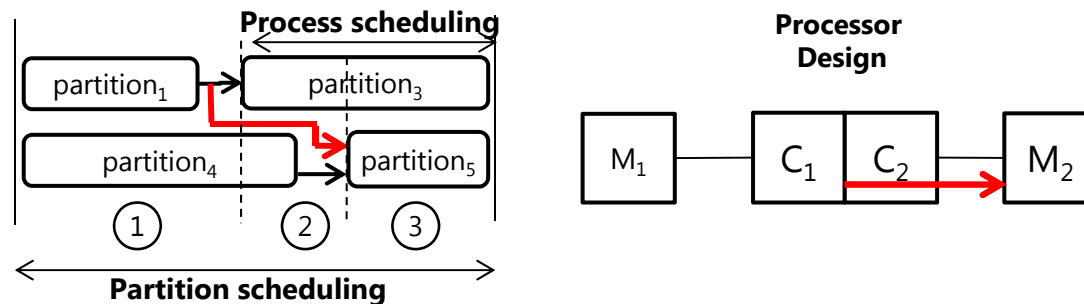


- Allows to compute the WCET estimation independently of other clusters
 - Provides bounds on NoC traversal
 - Hierarchal Rings: $2N-1$
 - MxN Mesh: between 3 and $3M + 2*(N-2)$
 - Provide bounds on memory access
 - Memory controller: $(\#Entries - 1) * t_{IL_{worst}}^{(1)}$

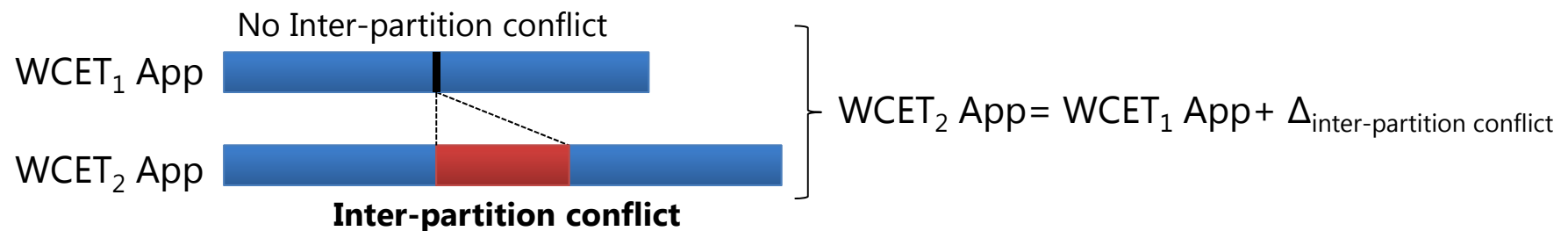


⁽¹⁾ Timing Effects of the Memory System in Real-Time Multicore Integrated Architectures: Problems and Solutions, ACM Transactions on Embedded Computing Systems. Marco Paolieri, Eduardo Quiñones, Francisco J. Cazorla, Mateo Valero (MERASA FP7 project)

- Inter-partition conflicts **impact** on cluster execution
 - The destination partition is still not executed

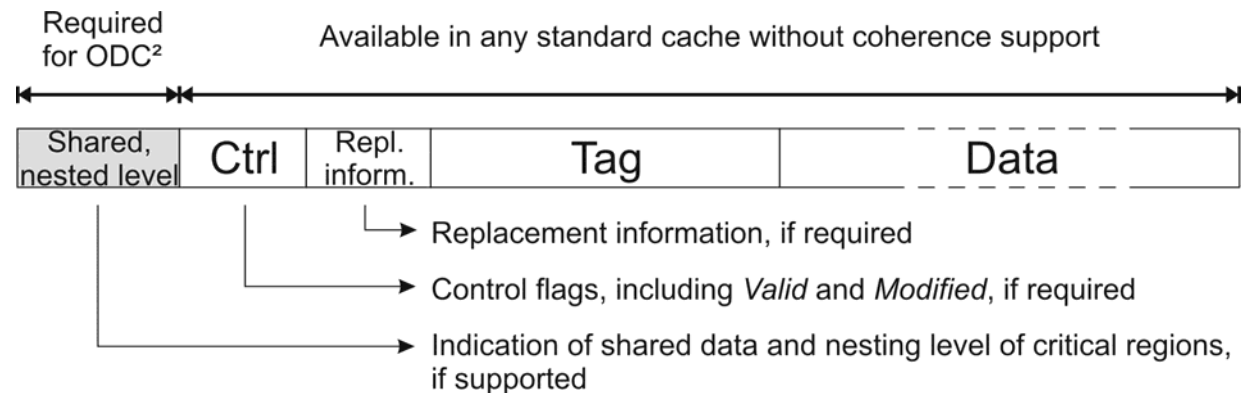


- Must be considered at **system integration**
 - The inter-partition communication is known at integration time



- Memory become a bottleneck when increasing the number of cores
 - Local cache per core necessary
 - Accesses to shared memory required because of large input data or shared data
- What's about consistency of shared memory accesses?
 - Need to keep all caches consistent/coherent
 - Means a predictable coherency protocol
- **On-demand consistency cache (ODC²)**
 - More details tomorrow on HIREs workshop...

- No cache coherency during *normal* program execution
- Consistency of memory accesses only when executing a critical section (locked by a mutex, semaphore, ...)
 - Every access to shared data is protected by a mutex (or similar)
- The cache coherence protocol guarantees that the shared data is within only one cache at any point in time
 - Only one application accesses to shared data at any point in time
- After exit the critical region shared data contained within the cache is invalidated



- Each cache line is equipped with a *shared data* flag (SDF)
 - will be extended to a *Nesting Level* (NL) information
- The SDF/NL is set at every cache miss between an *entry/exit*
 - to be honest: NL will be set on every access according to the actual level of nested critical sections
- At an *exit*, all cache lines with an active SDF or with NL > *new nested level* are
 - Written back, if required (write-through technique preferred between entry/exit)
 - Invalidated
- After an exit, there is no shared data of that level available in the cache

- Timing isolation properties to parallel hard real-time application running on many-core architectures
 - Parallel Software Partitions
 - Provided at software level
 - Guaranteed Resource Partitions
 - Provided at hardware level
 - Facilitates de allocation algorithm
- Time-aware hardware mechanisms
 - Design of Network on Chip and Memory
 - Clusterised architectures
 - Predictable Cache Coherence Mechanisms

parMERASA

Many-core Processor Architecture

Dr. Eduardo Quiñones

Dr. Sascha Uhrig

