Analysis and Optimization of Mixed-Criticality Applications on Partitioned Distributed Architectures

Domițian Tămaș-Selicean, Sorin Ovidiu Marinescu and Paul Pop

Technical University of Denmark

recomp

Reduced Certification Costs for trusted Multi-core Platforms

DTU Informatics

Department of Informatics and Mathematical Modeling

Outline

- Motivation
- Separation of mixed-criticality applications
  - At processing element level
  - At communication level
- Problem formulation and example
- Optimization strategy
- Experimental results
- Conclusions

Motivation

- **Safety** is the property of a system that will not endanger human life or the environment
- A safety-related system needs to be **certified**

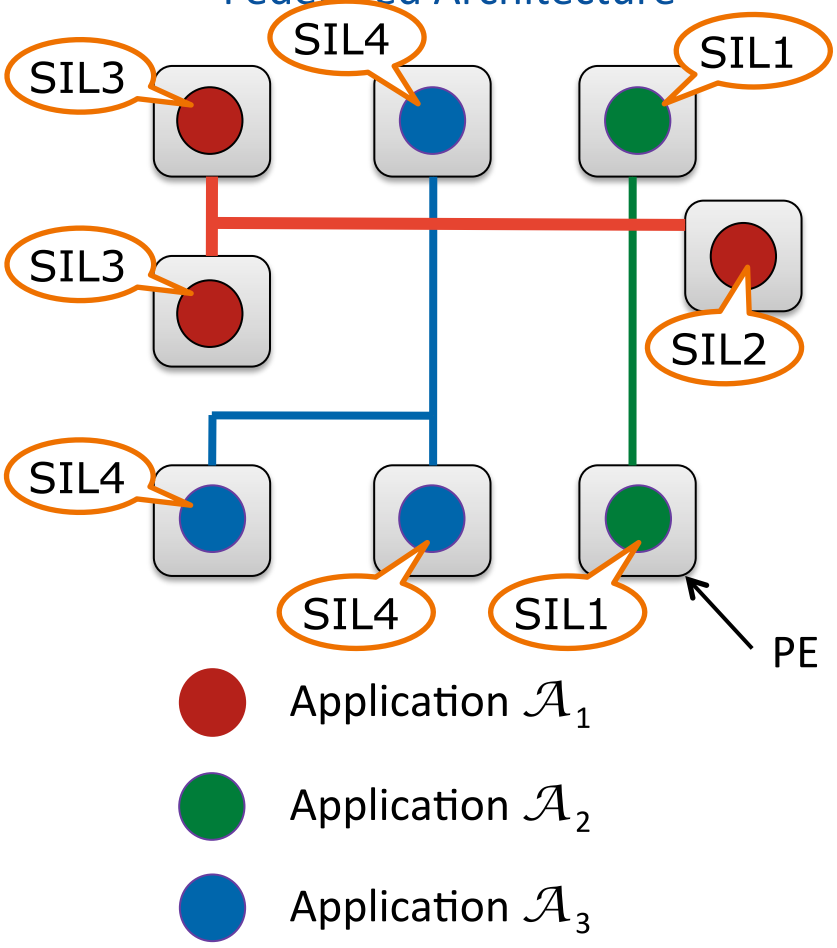
- A Safety Integrity Level (SIL) is assigned to each safety related function, depending on the required level of risk reduction
- There are 4 SILs:
  - SIL4 (most critical)
  - SIL1 (least critical)
  - SIL0 (non-critical) – not covered by standards

- SILs dictate the development process and certification procedures

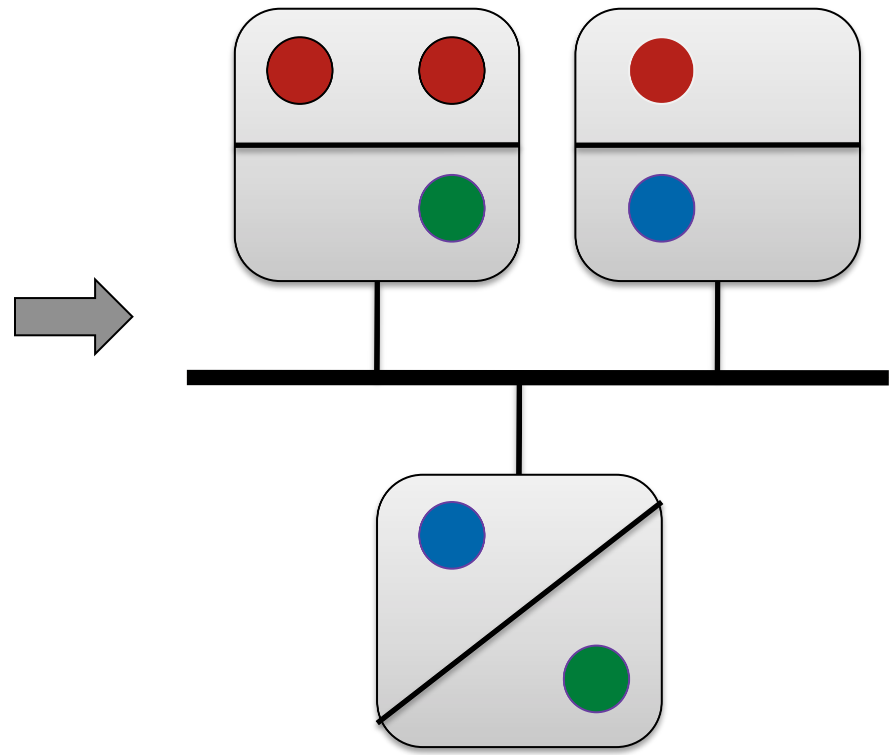
Motivation

- Real time applications implemented using distributed systems
- Mixed-criticality applications share the same architecture

Federated Architecture

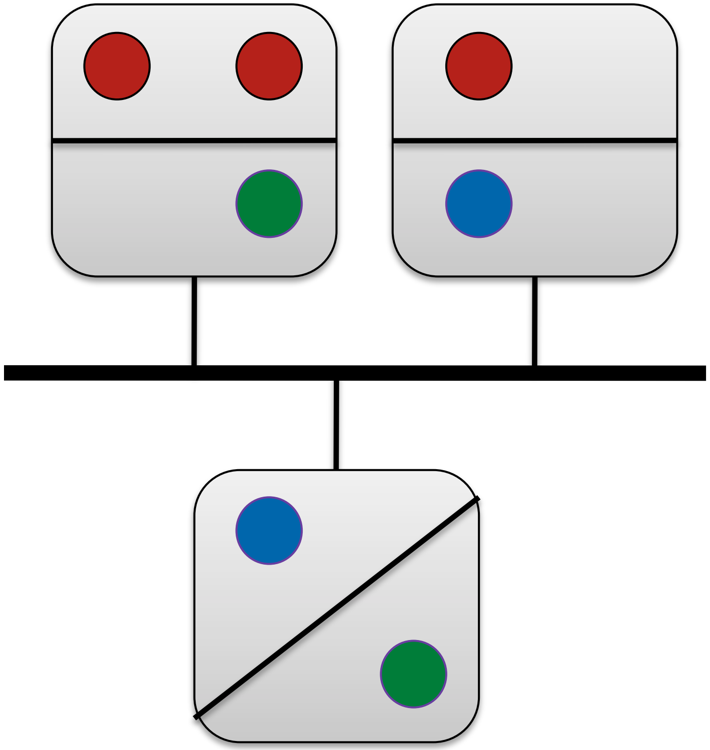


Integrated Architecture



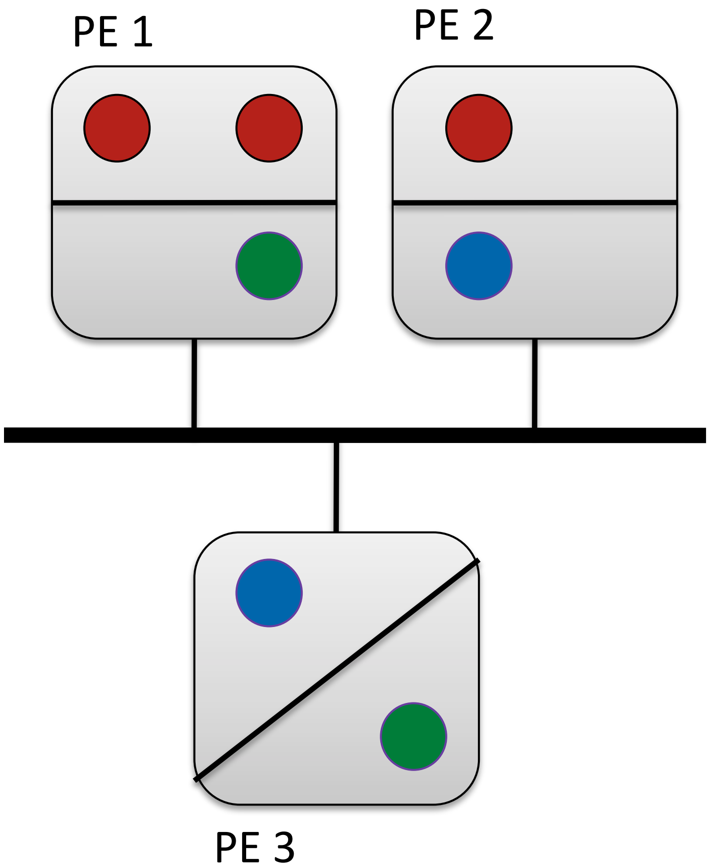
Solution: partitioned architecture

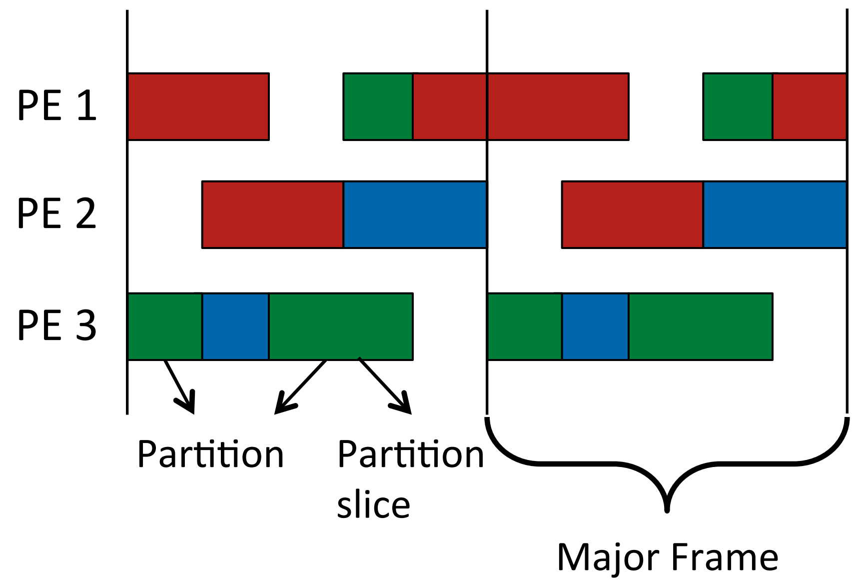
Separation at PE-level



- Partition = virtual dedicated machine
- Partitioned architecture
  - Spatial partitioning
    - protects one application's memory and access to resources from another application
  - Temporal partitioning
    - partitions the CPU time among applications

Separation at PE-level

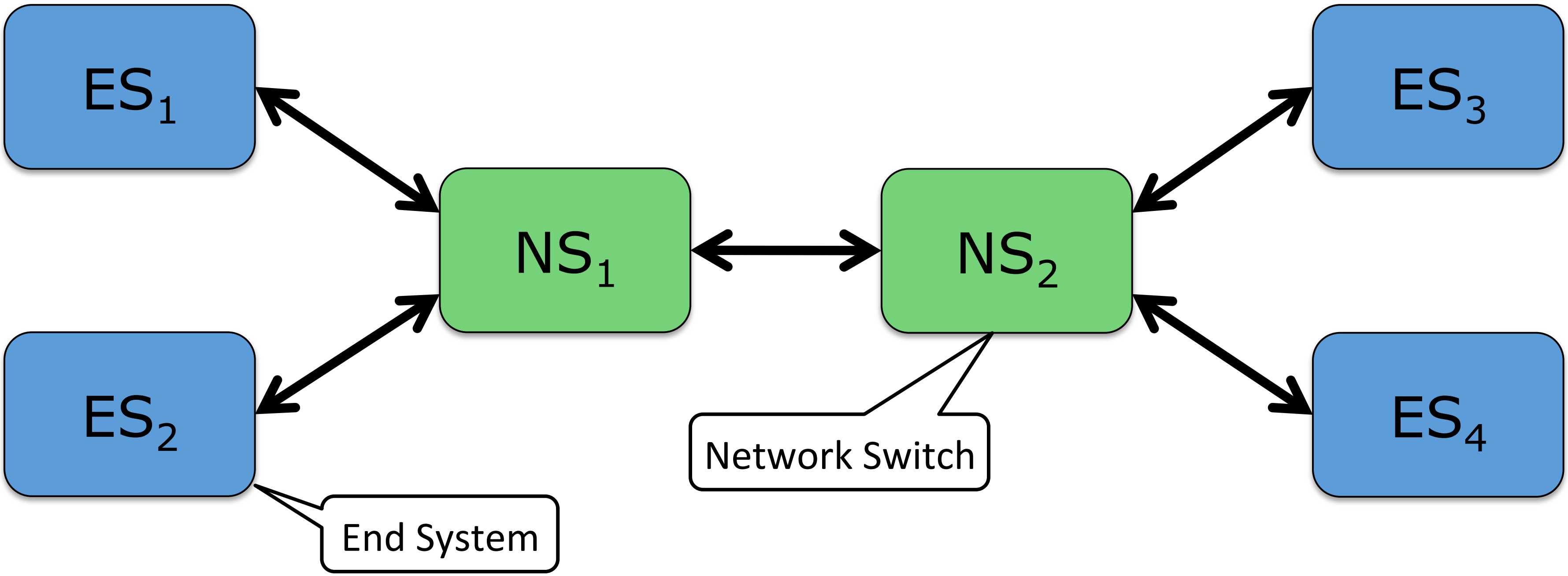




- Temporal partitioning
  - Static partiti
    - Rep eriod MF
    - itch overhead
    - h partition can have its own scheduling policy
    - A partition has a certain SIL

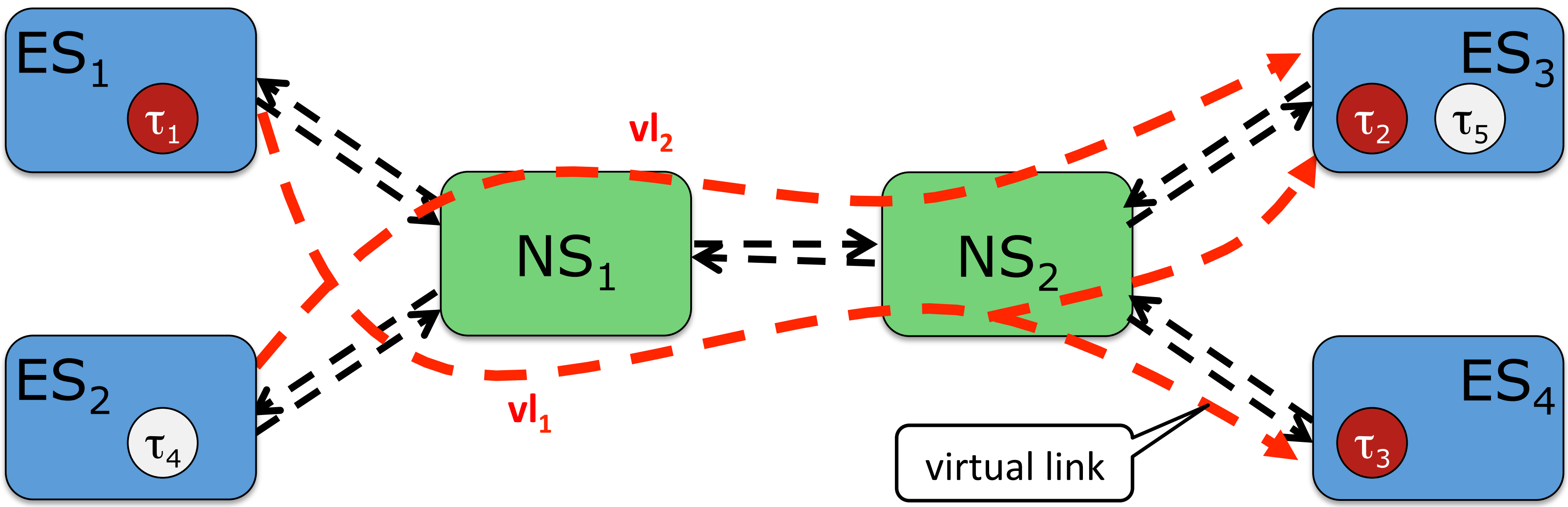
Problem: optimize task mapping and allocation of partitions

Separation at Network-level



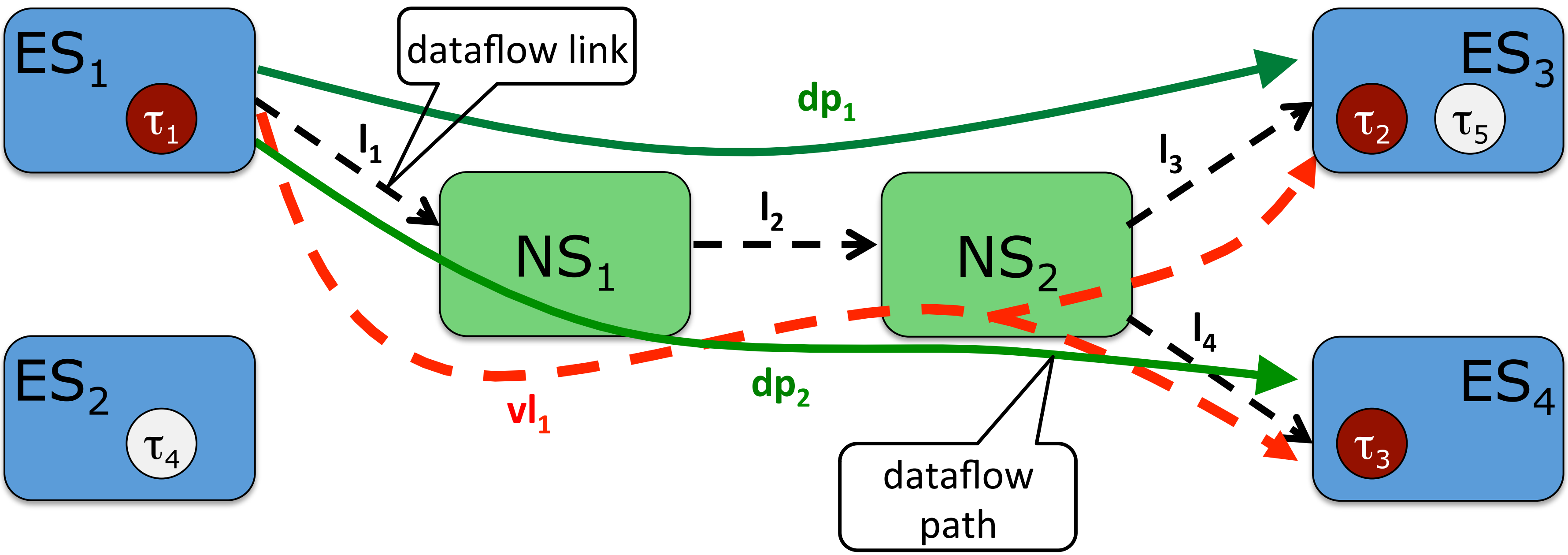
- Full-Duplex Ethernet-based data network for safety-critical applications
- Compliant with ARINC 664p7 "Aircraft Data Network"

Separation at Network-level



- Highly critical application $\mathcal{A}_1$: τ_1, τ_2 and τ_3
  - τ_1 sends message m_1 to τ_2 and τ_3
- Non-critical application $\mathcal{A}_2$: τ_4 and τ_5
  - τ_4 sends message m_2 to τ_5

Separation at Network-level

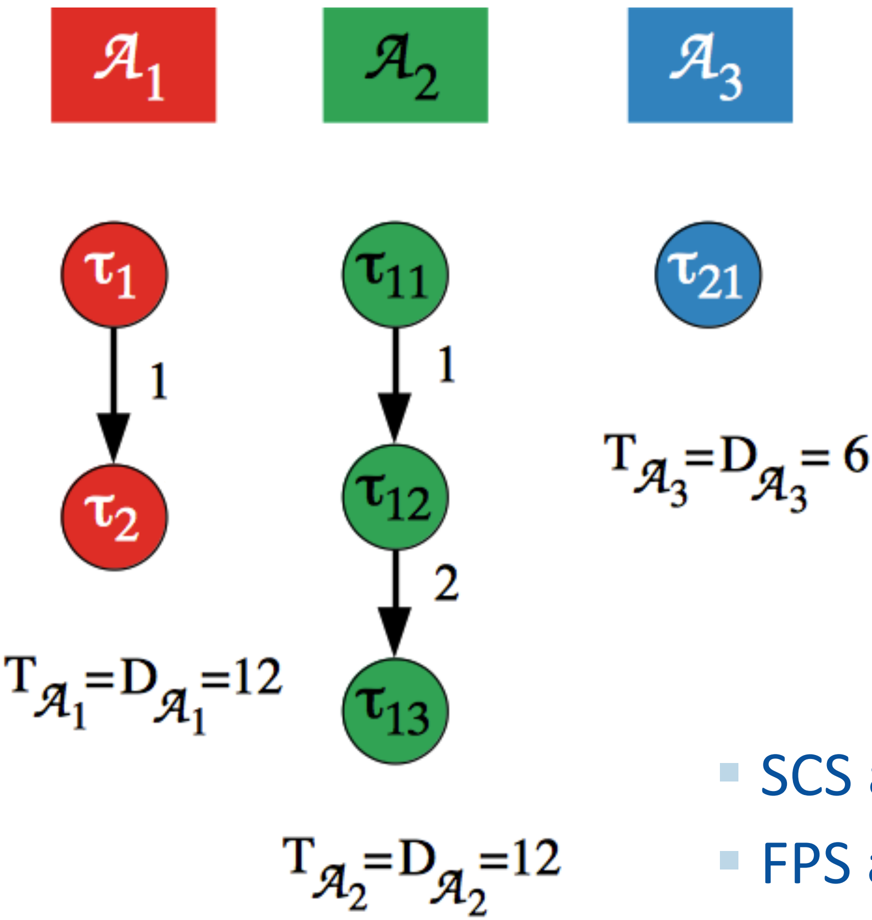


- Highly critical application $\mathcal{A}_1$: τ_1, τ_2 and τ_3
  - τ_1 sends message m_1 to τ_2 and τ_3
- Non-critical application $\mathcal{A}_2$: τ_4 and τ_5
  - τ_4 sends message m_2 to τ_5

TTEthernet

- Traffic classes
  - Time Triggered (TT)
    - based on static schedule tables
  - Rate Constrained (RC)
    - deterministic unsynchronized communication
    - ARINC 664p7 traffic
  - Best Effort (BE)
    - no timing guarantees provided

Application Model



	$\mathcal{A}_1$		$\mathcal{A}_2$			$\mathcal{A}_3$
	τ_1	τ_2	τ_{11}	τ_{12}	τ_{13}	τ_{21}
N_1	2	x	2	3	3	1
N_2	4	5	3	5	4	2

WCET and mapping restrictions

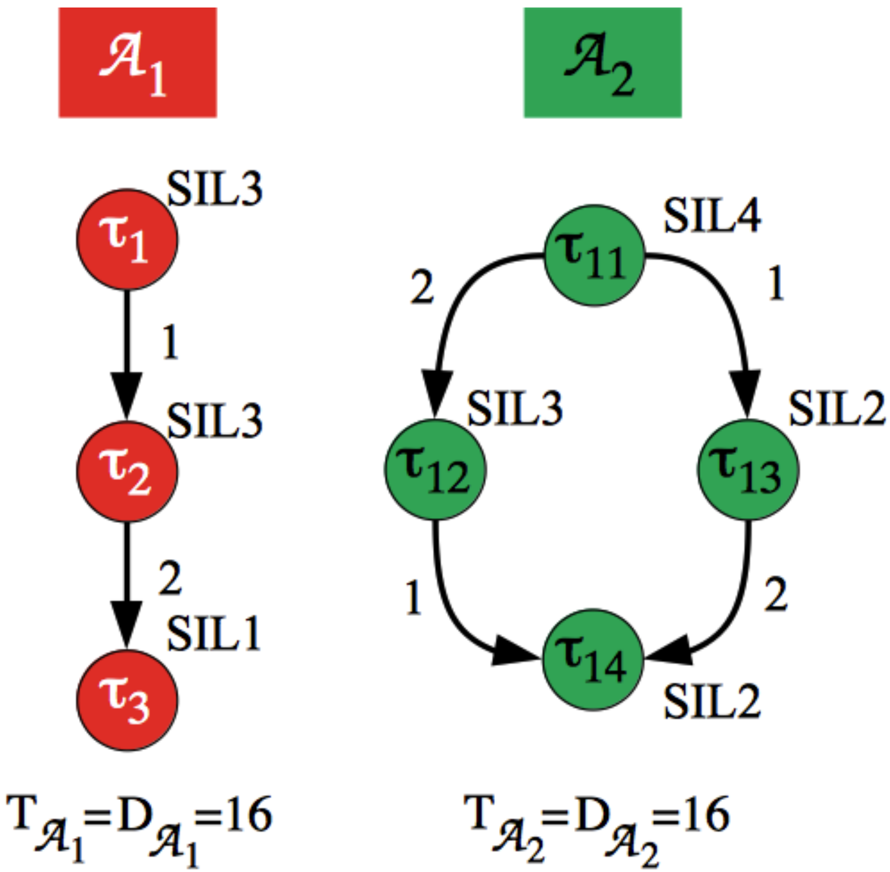
- SCS apps transmit TT messages
- FPS apps transmit RC messages

Problem formulation

- Given
  - A set of applications
  - The criticality level (or SIL) of each task
  - A set of N processing elements (PEs) and topology of the network
  - The set of TT and RC frames
  - The set of virtual links
  - The size of the Major Frame and of the Application Cycle
- Determine
  - The mapping of tasks to PEs
  - The sequence and length of partition slices on each processor
  - The assignment of tasks to partitions
  - The schedule for all the tasks and TT frames in the system
- Such that
  - All applications meet their deadline
  - The response times of the FPS tasks and RC frames is minimized

Motivational Example 1

- Mapping and partitioning optimization

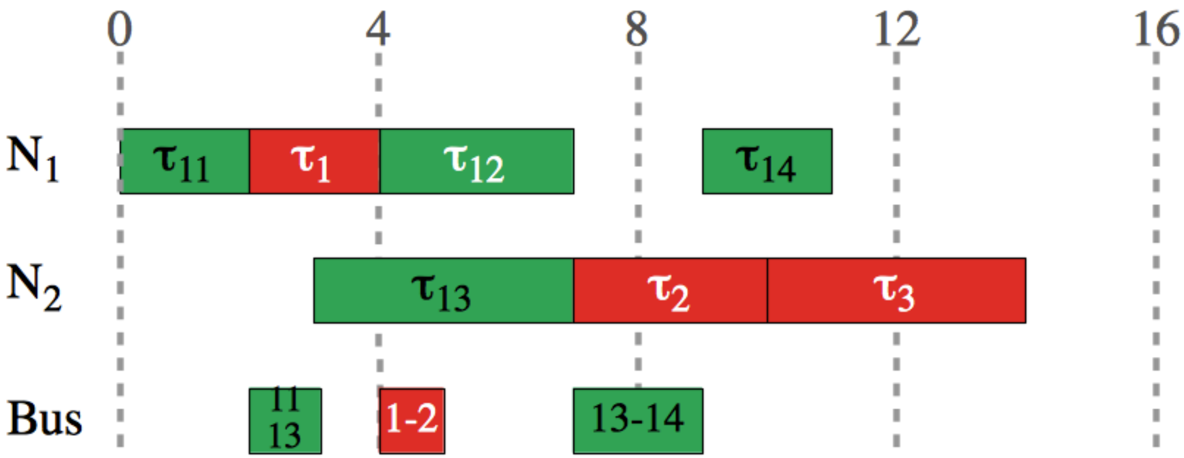


Mixed-criticality applications

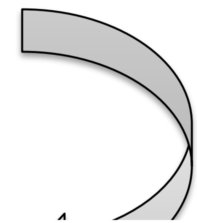
		N_1	N_2
$\mathcal{A}_1$	τ_1	2	4
	τ_2	x	3
	τ_3	3	4
$\mathcal{A}_2$	τ_{11}	2	3
	τ_{12}	3	5
	τ_{13}	x	4
	τ_{14}	2	3

WCET and mapping restrictions

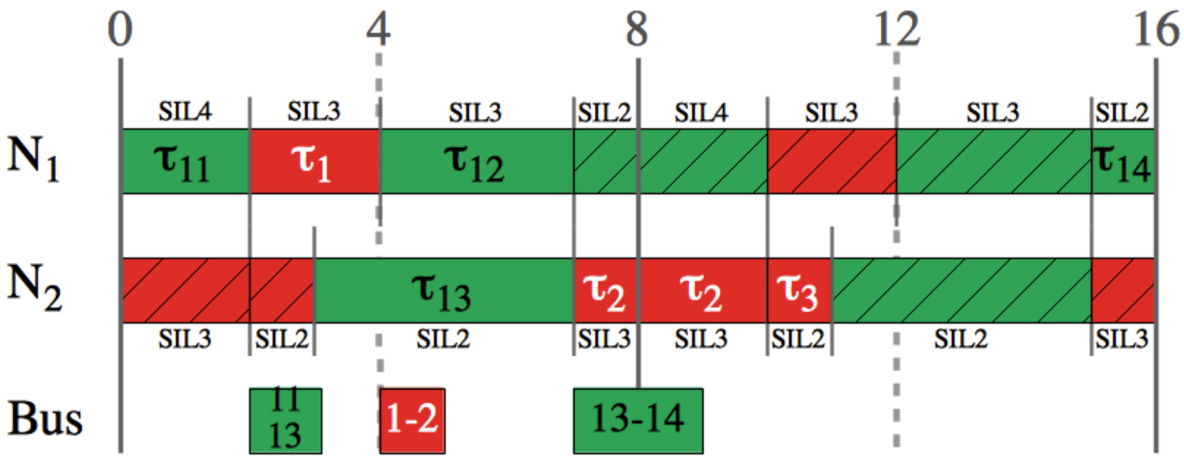
Motivational Example 1



Optimal mapping, without considering partitions.



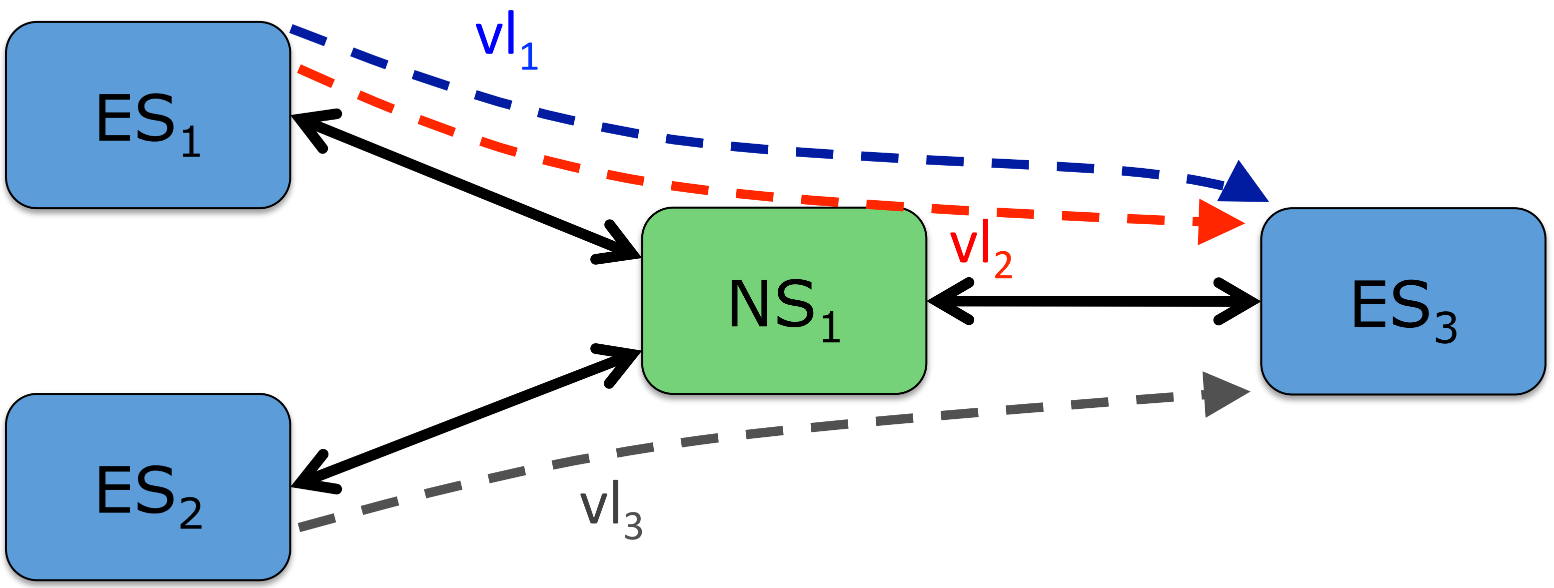
Motivational Example 1



Partitioning, using the previously obtained mapping. τ_3 and τ_{14} miss their deadline.

Motivational Example 2

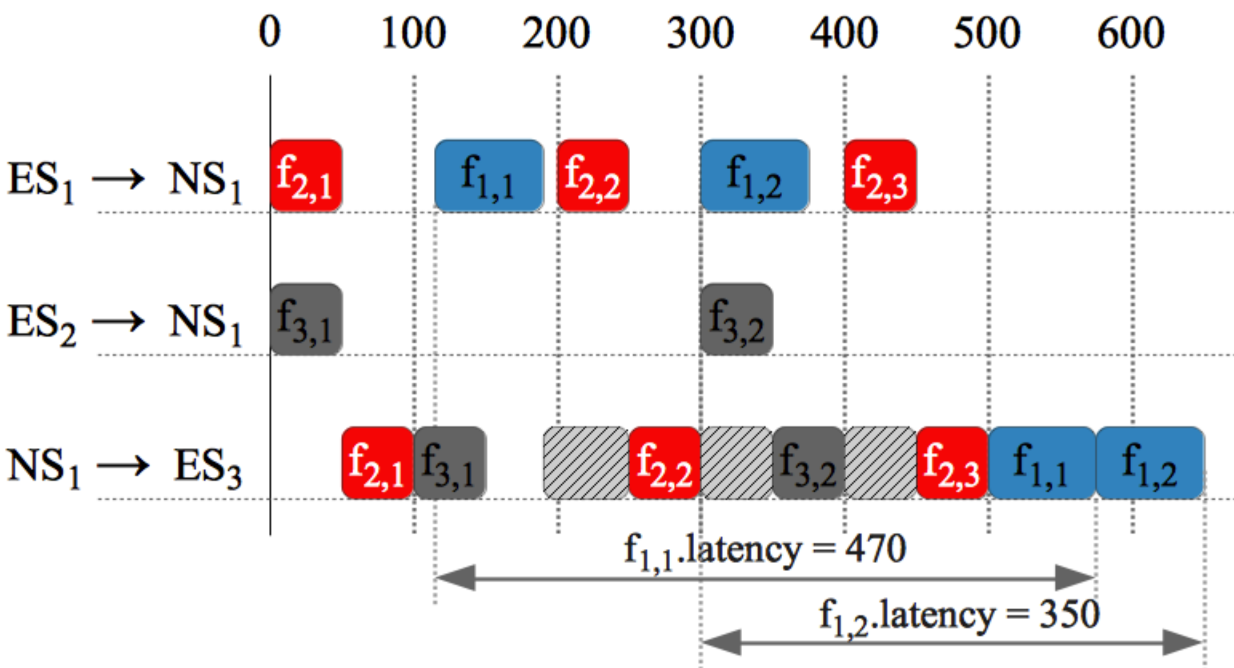
- Optimization of TT message schedules

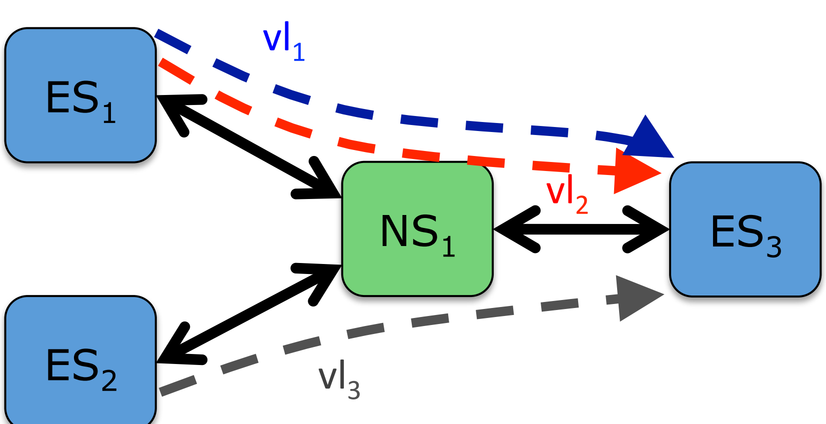


	period (us)	deadline (us)	C_i (us)	$\mathcal{M}$
f1 $\in \mathcal{F}^{RC}$	300	300	75	vl_1
f2 $\in \mathcal{F}^{TT}$	200	200	50	vl_2
f3 $\in \mathcal{F}^{TT}$	300	300	50	vl_3

Motivational Example 2

- Initial TT schedule

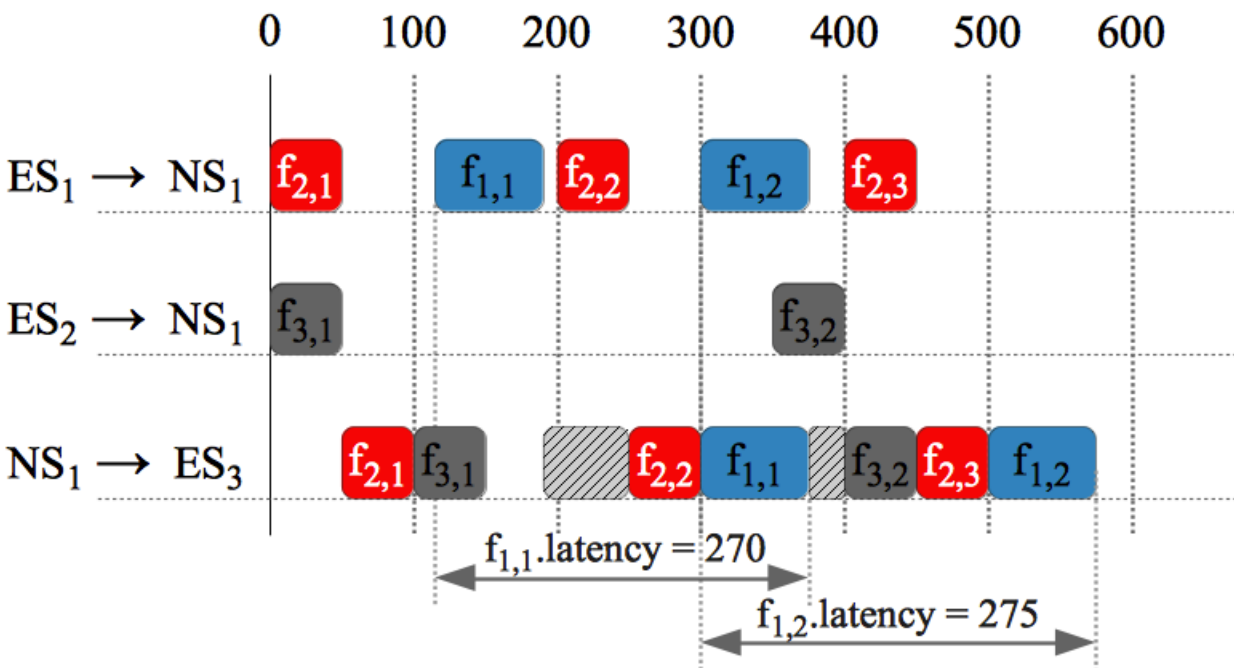


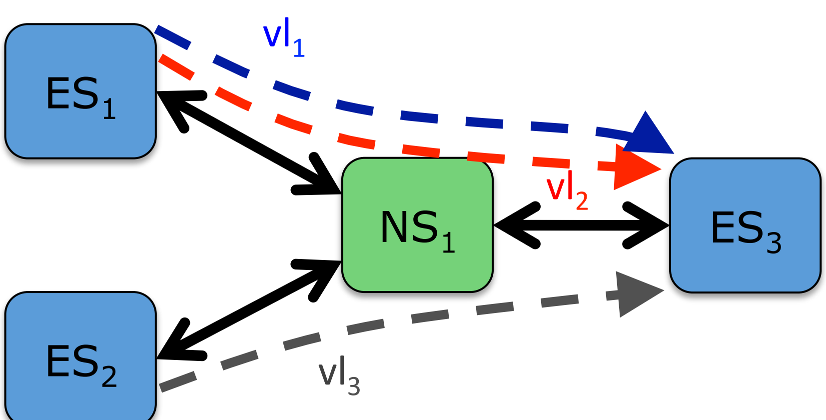


	period (us)	deadline (us)	C_i (us)	$\mathcal{M}$
f1 $\in \mathcal{F}^{RC}$	300	300	75	vl_1
f2 $\in \mathcal{F}^{TT}$	200	200	50	vl_2
f3 $\in \mathcal{F}^{TT}$	300	300	50	vl_3

Motivational Example 2

- Optimized TT schedule





	period (us)	deadline (us)	C_i (us)	$\mathcal{M}$
f1 $\in \mathcal{F}^{RC}$	300	300	75	vl_1
f2 $\in \mathcal{F}^{TT}$	200	200	50	vl_2
f3 $\in \mathcal{F}^{TT}$	300	300	50	vl_3

Optimization Strategy

- Tabu Search meta-heuristic
  - Task mapping and partition slice optimization (TO)
    - Considering TT frame schedules fixed
  - TT frame schedules optimization (TM)
    - Considering the task mapping and partition slices fixed

- Tabu Search
  - Minimizes the cost function
  - Explores the solution space using design transformations

Optimization Strategy

- Degree of schedulability
  - Captures the difference between the worst-case response time and the deadline

- Cost Function

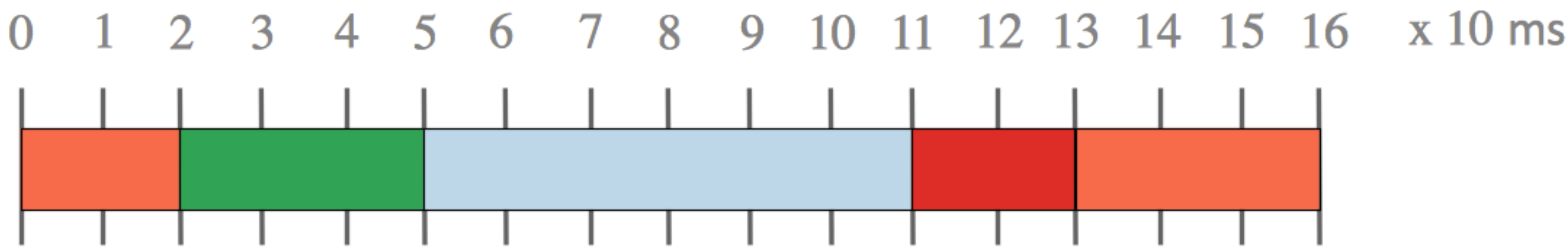
$$Cost(\Psi) = \begin{cases} c_1 = \sum_{\mathcal{A}_i \in \Gamma} \max(0, R_i - D_i) & if c_1 > 0 \\ c_2 = \sum_{\mathcal{A}_i \in \Gamma} (R_i - D_i) & if c_1 = 0 \end{cases}$$

Optimization Strategy: Design Transformations

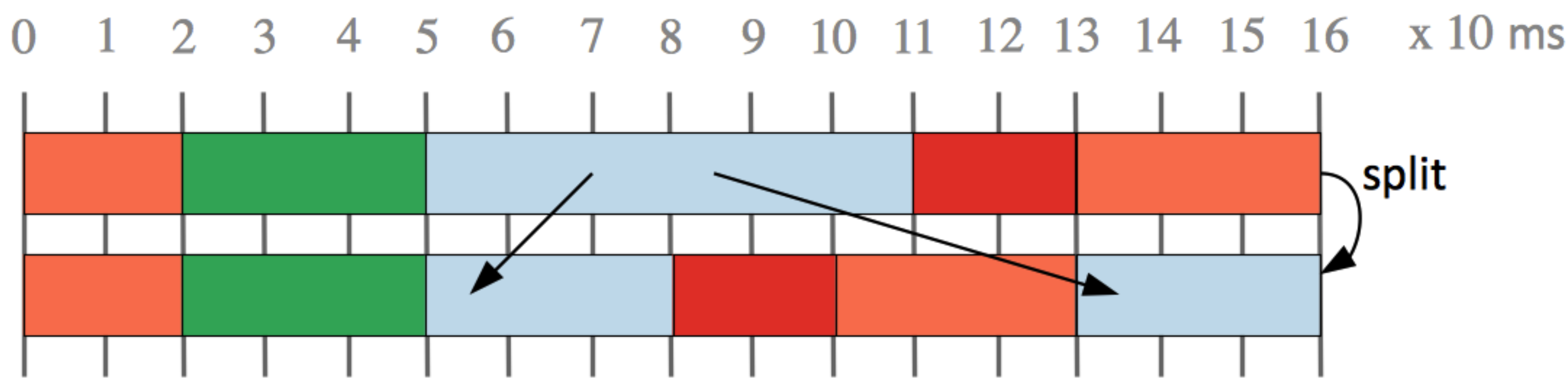
- Partition slice moves
  - resize partition slice
  - swap two partition slices
  - join two partition slices
  - split partition slice into two

- Task moves
  - re-assign task to another partition

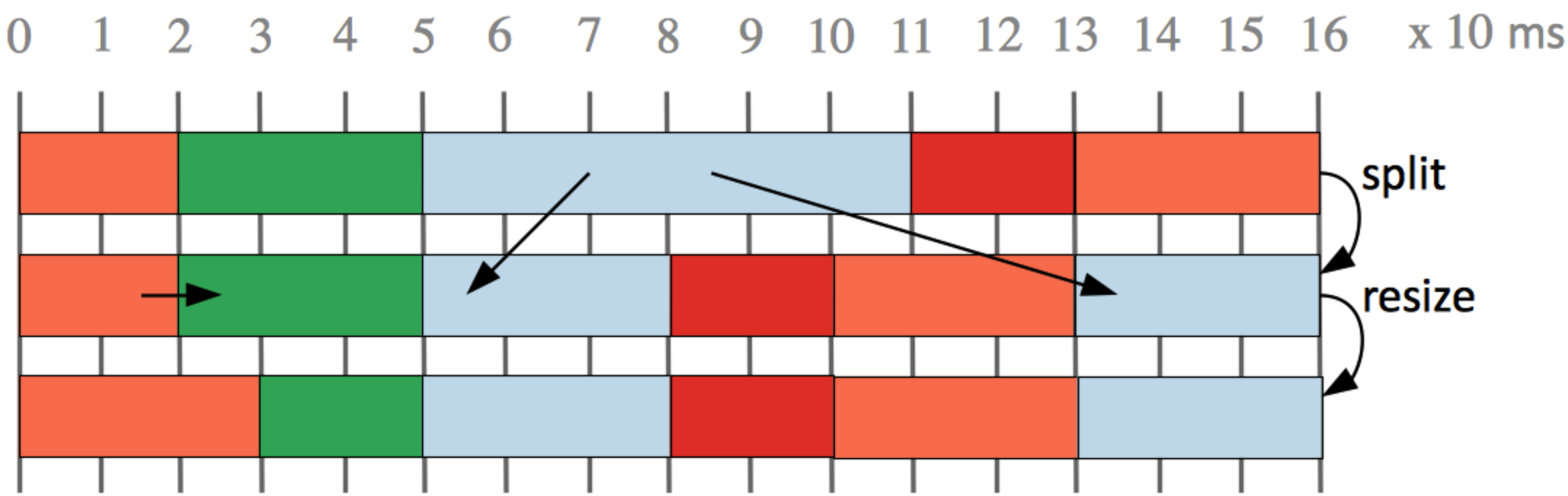
Optimization Strategy: Design Transformations



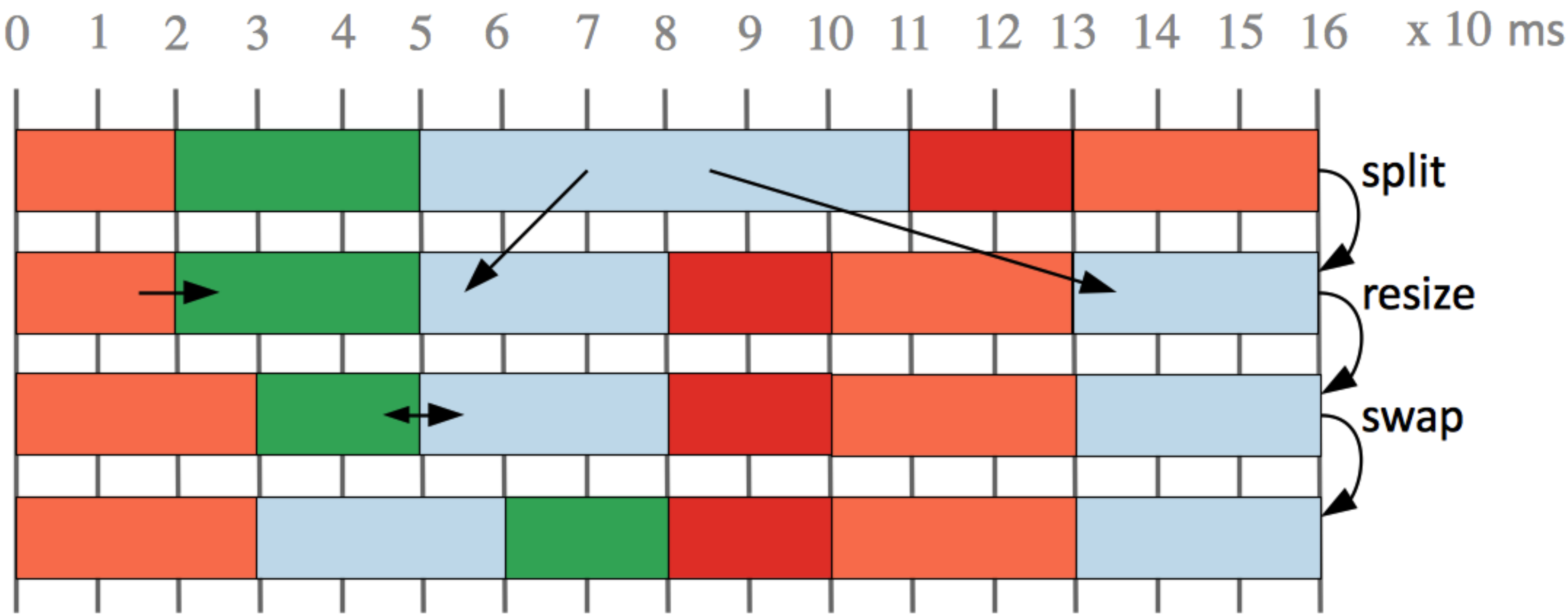
Optimization Strategy: Design Transformations



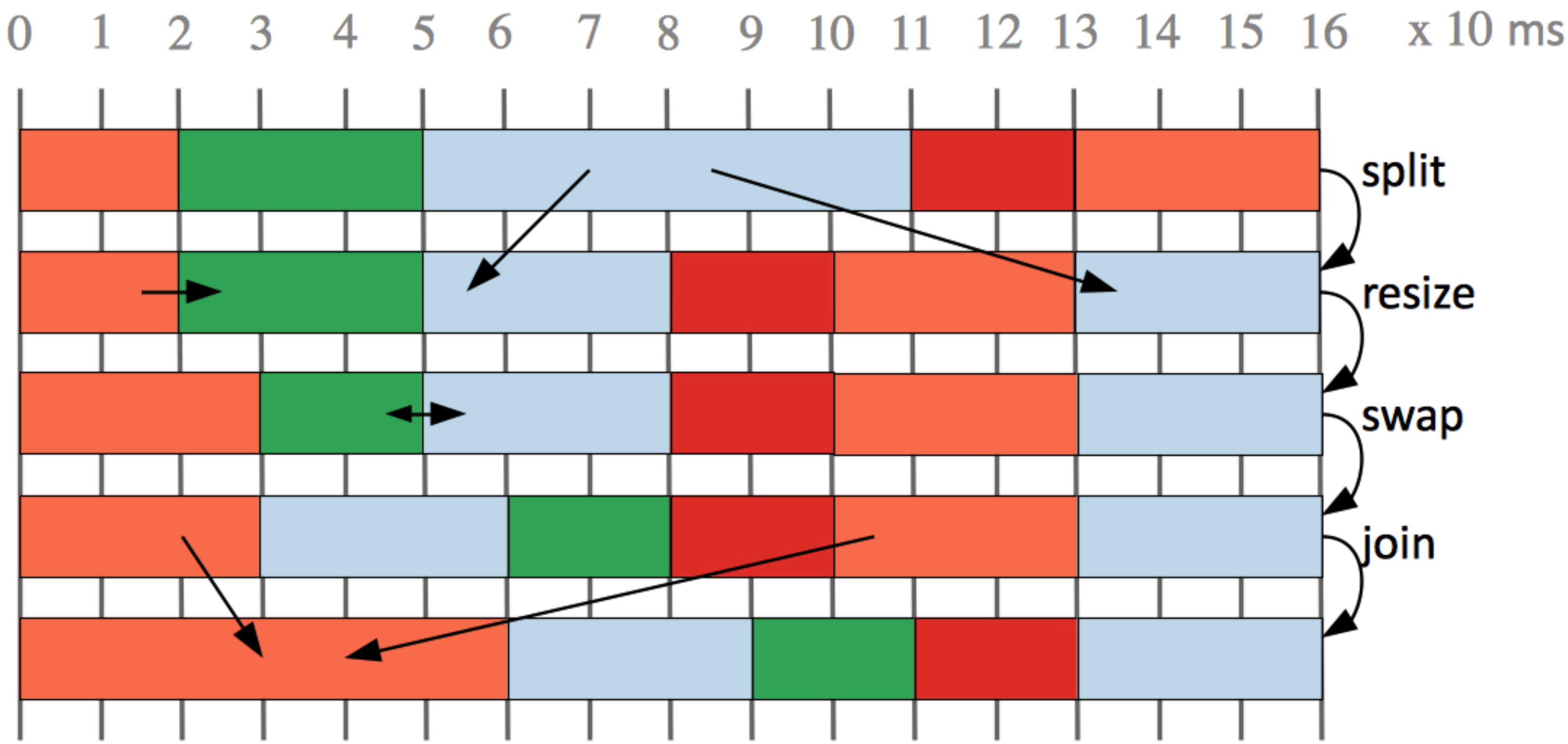
Optimization Strategy: Design Transformations



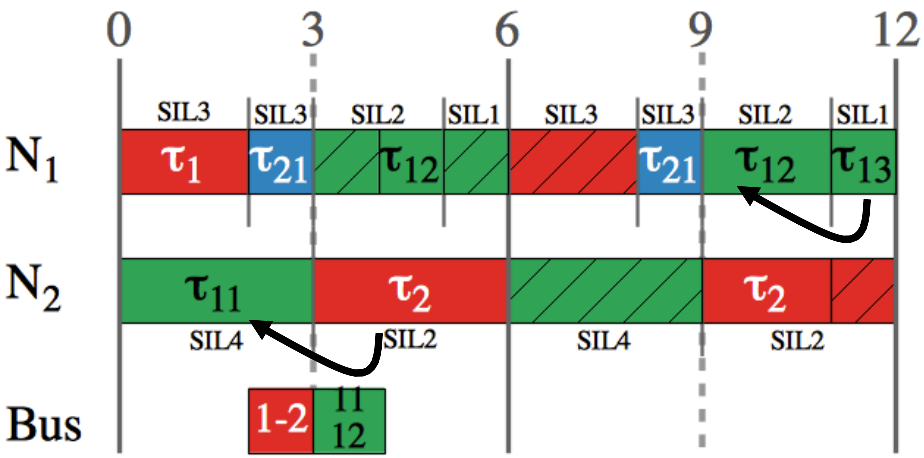
Optimization Strategy: Design Transformations

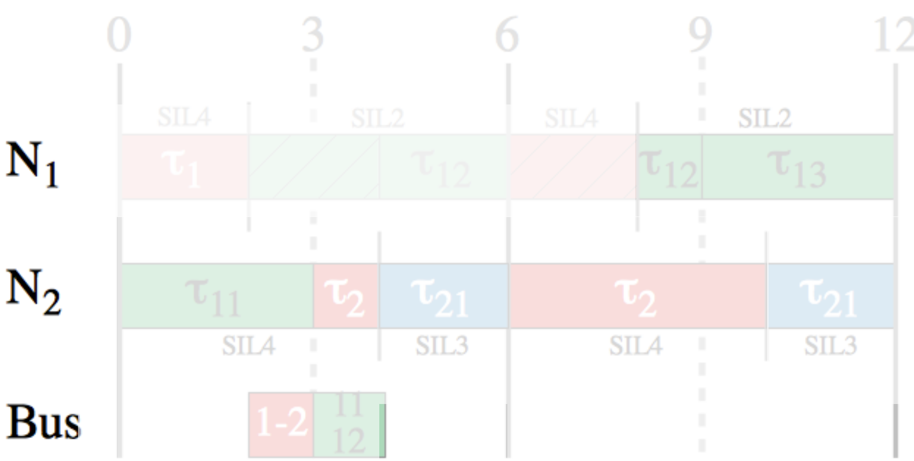


Optimization Strategy: Design Transformations



Optimization Strategy: Design Transformation





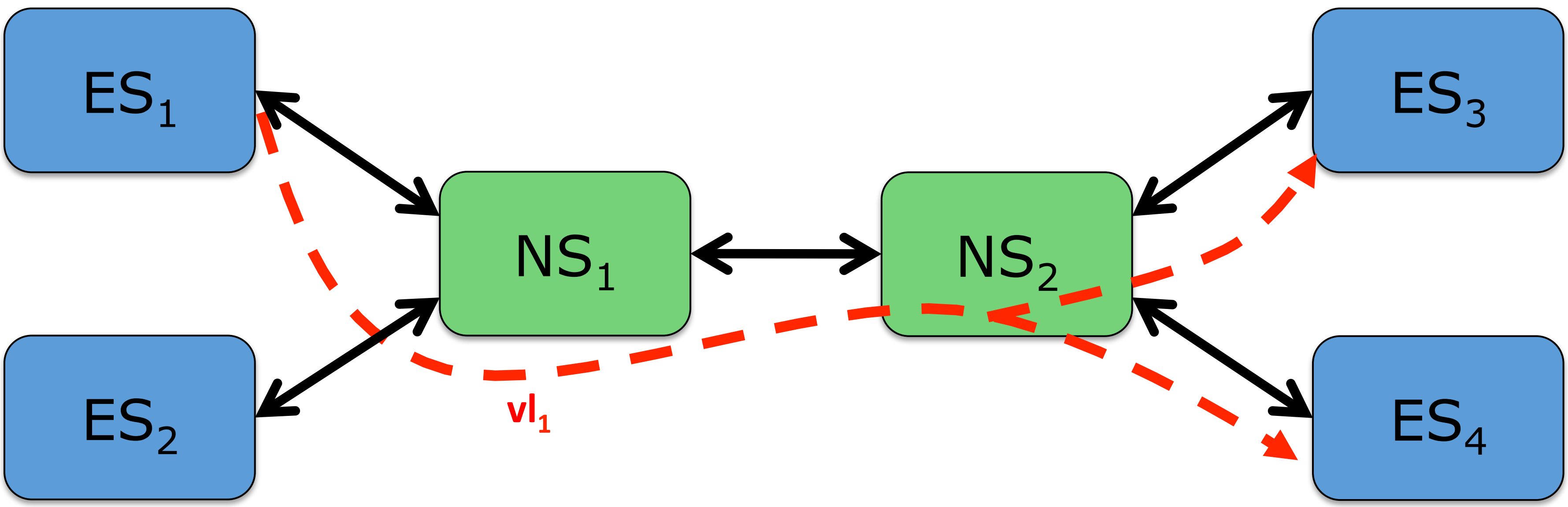
- Task **re-assignment** move
  - To another partition of the same application
  - To a partition of another application
  - To a newly created partition
  - Empty partitions are deleted

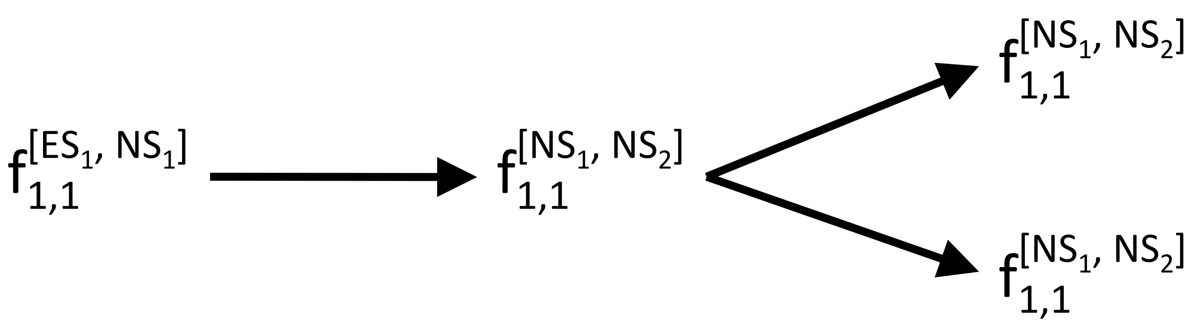
Optimization Strategy: Design Transformations

- TT frame moves
  - advance frame transmission time
  - advance frame predecessors transmission time
  - postpone frame transmission time
  - postpone frame successors transmission time

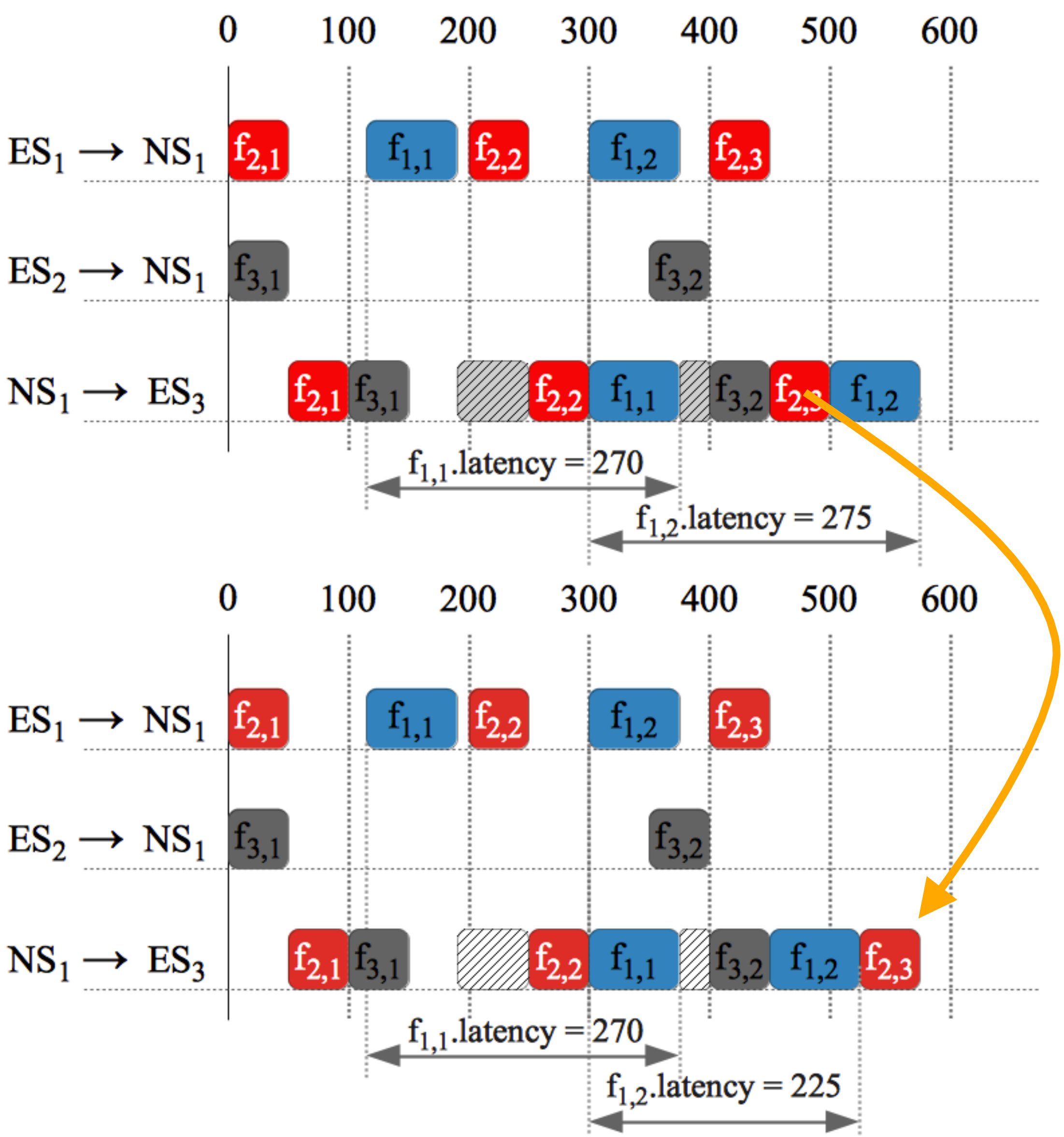
- RC frame moves
  - reserve space for RC frame
  - resize reserved space for RC frame
  - remove reserved space for RC frame

Frame Representation for Moves

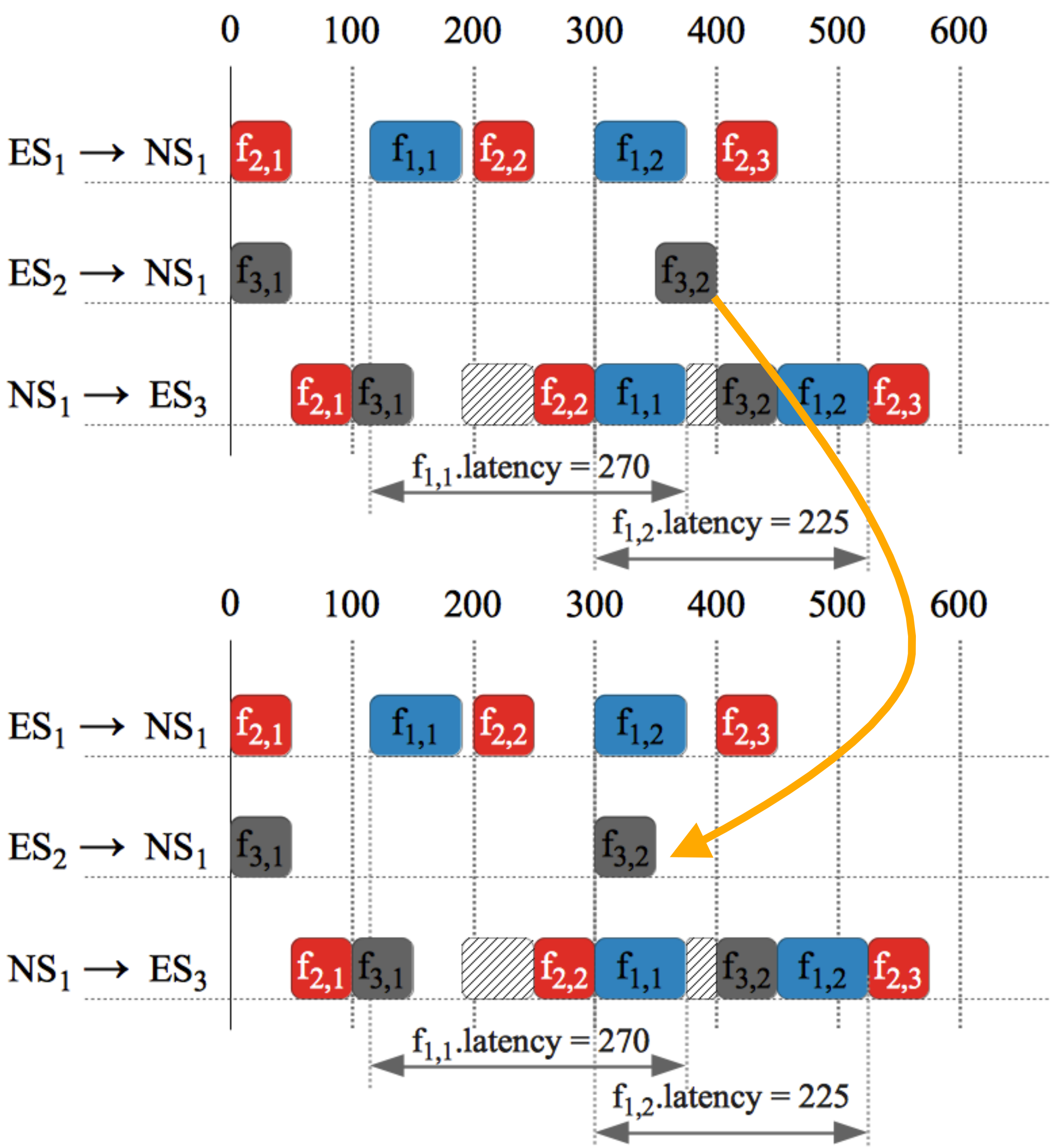




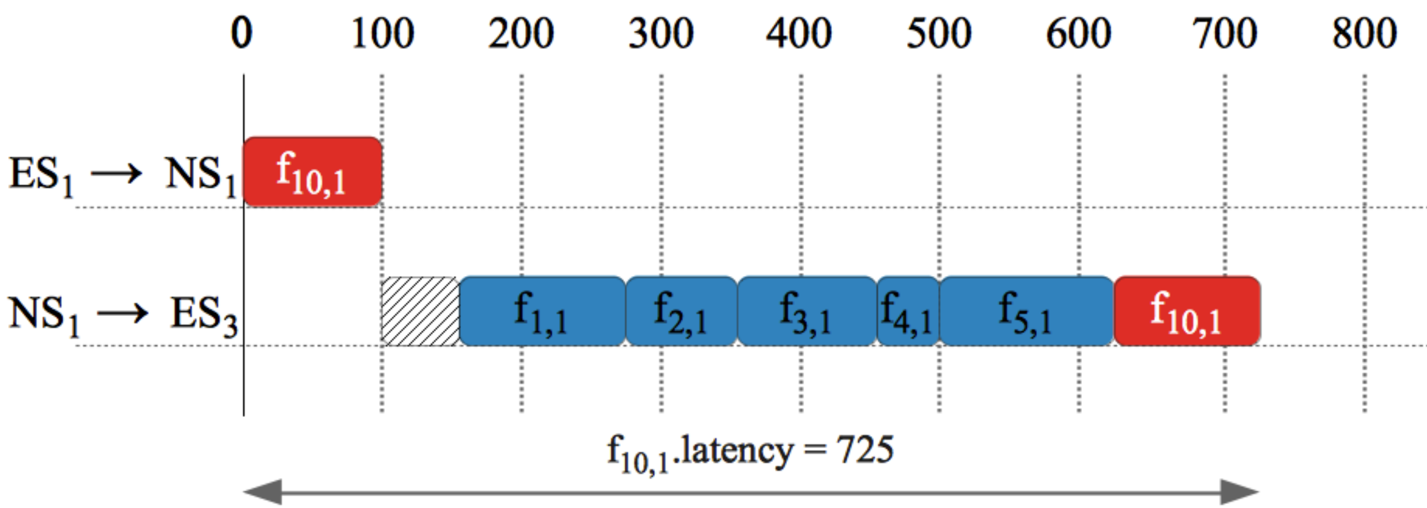
Design transformations: Postpone move

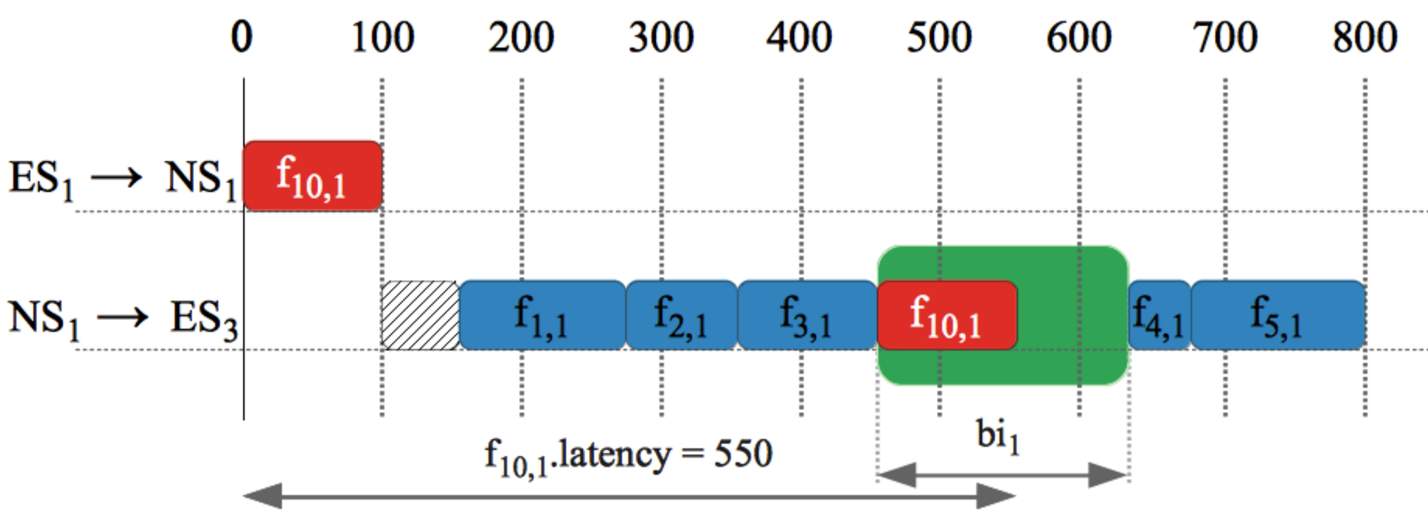


Design transformations: Advance move

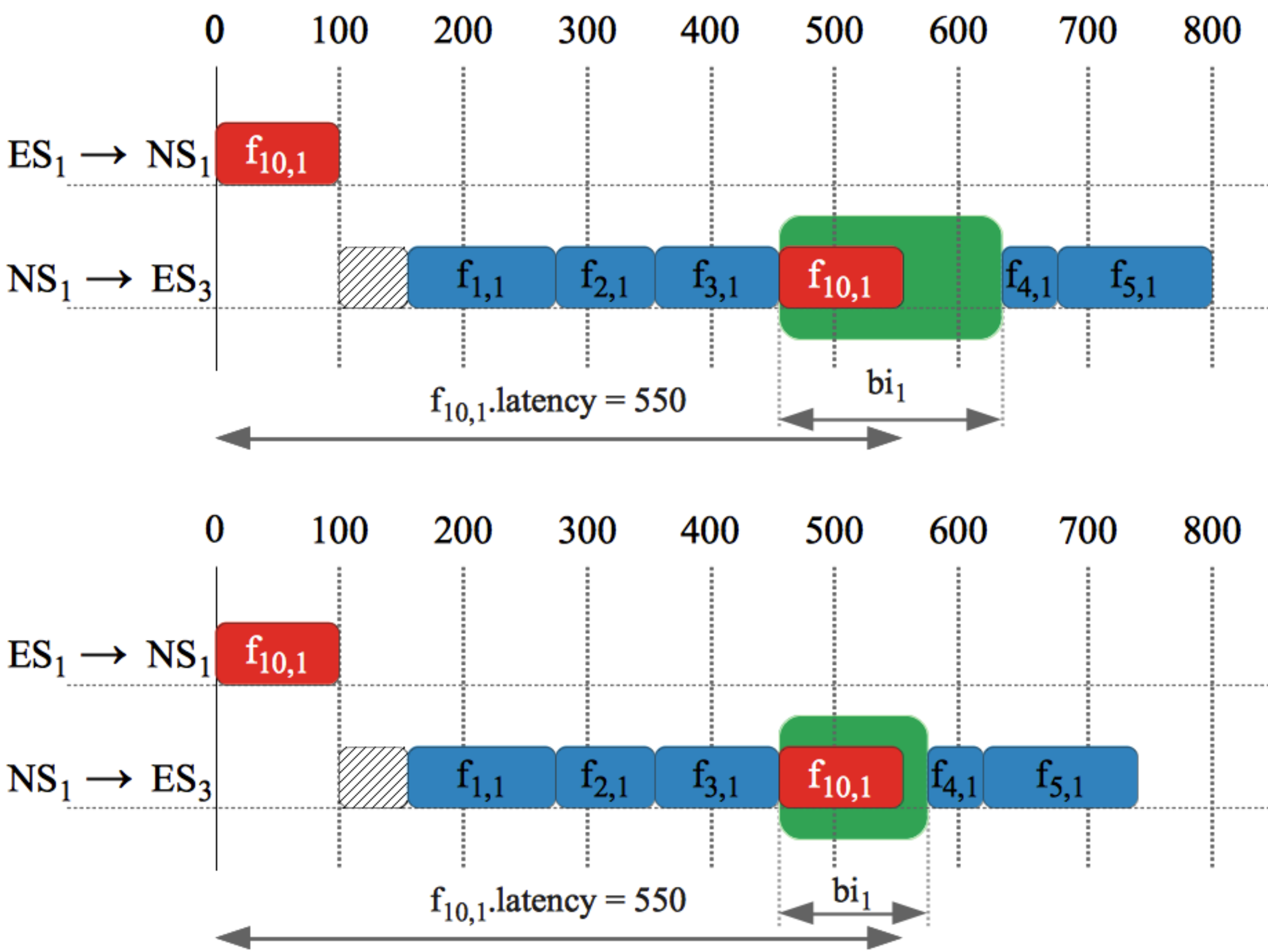


Design transformations: Reserve space for RC





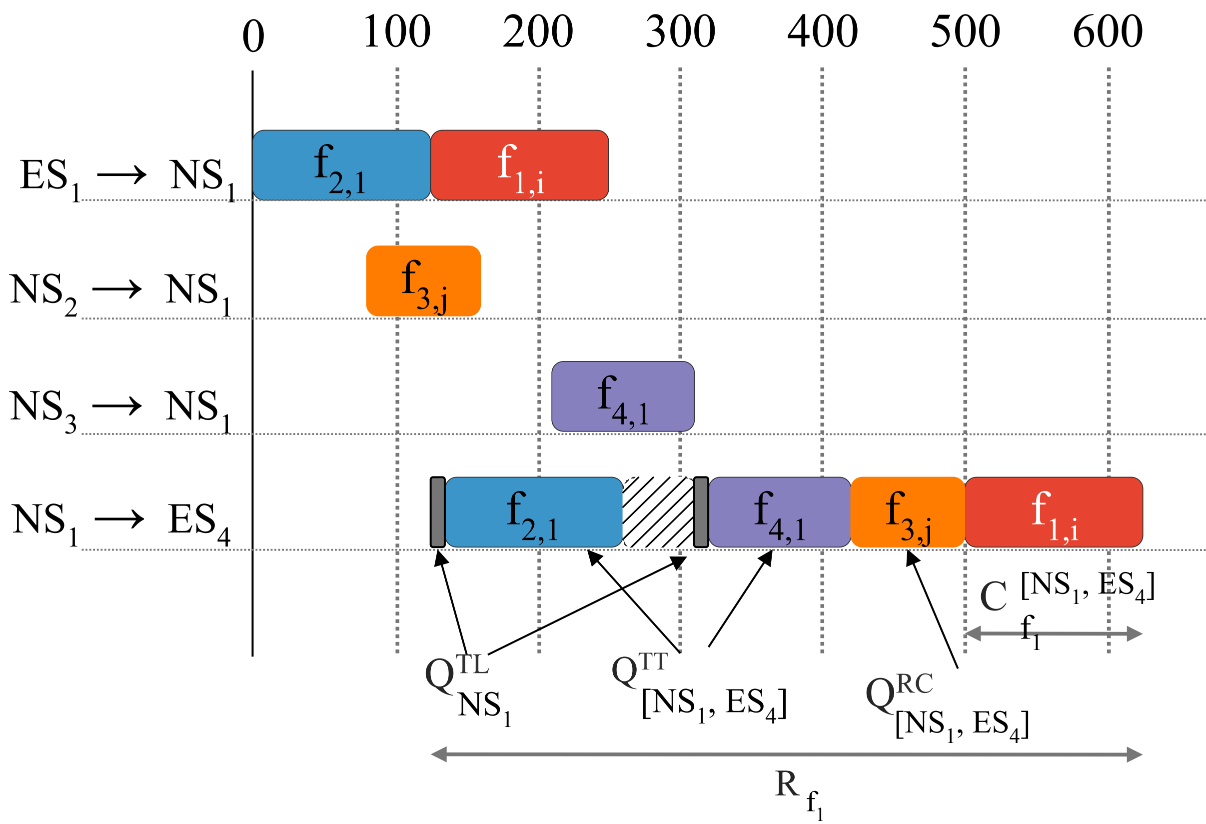
Design transformations: Resize RC reserved space

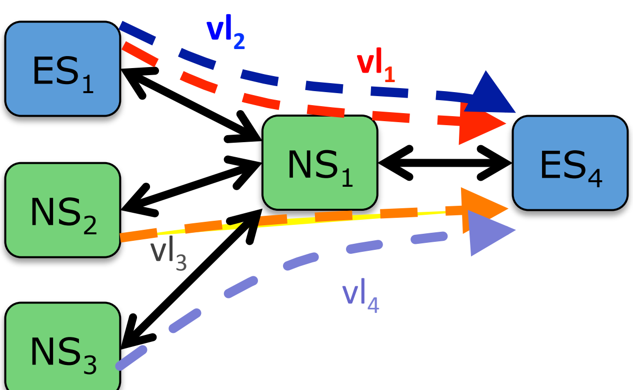


RC Frame End-to-End Analysis

- On a dataflow link, a RC frame can be delayed by:
  - scheduled TT frames
  - queued RC frames
  - technical latency
  - policy specific:
    - timely block
    - pre-emption

RC Frame End-to-End Analysis





$$Q_{f_i}^{[\nu_j,\nu_k]} = Q_{f_i,[\nu_j,\nu_k]}^{TT} + Q_{f_i,[\nu_j,\nu_k]}^{RC} + Q_{\nu_j}^{TL}$$

$$R_{f_i} = \sum_{\substack{\nu_j,\nu_k \in \mathcal{V} \\ [\nu_j,\nu_k] \in vl_i}} \left(Q_{f_i}^{[\nu_j,\nu_k]} + C_{f_1}^{[\nu_j,\nu_k]}\right)$$

RC Frame End-to-End Analysis

- Approaches for analysis of ARINC 644p7 network traffic:
  - Network Calculus, (Boyer, 2008)
  - Finite State Machine, (Saha, 2007)
  - Timed Automata, (Adnan, 2010)
  - Trajectory Approach, (Bauer, 2009)

- We use the method proposed in (Steiner, 2011)
  - it takes into account also the TT traffic
  - it is pessimistic:
    - does not ignore frames that already delayed a RC frame on a previous link
    - assumes uniformly distributed intervals of equal length reserved for RC traffic

Experimental Results: TO

- Benchmarks
  - 5 synthetic
  - 2 real life test cases from E3S
- TO compared to:
  - Straightforward Solution for Tasks (SST)
    - Simple partitioning scheme, each application $\mathcal{A}_i$ is allocated a total time proportional to the utilization of tasks of $\mathcal{A}_i$ on the processor they are mapped to

Experimental Results: TO

Set	Tasks	PEs	SST Sched. Tasks	TO Sched. Tasks	avg. % increase in δ
1	20	2	10	All	832.88
	26	3	13	All	27.36
	40	4	6	All	88.41
	50	5	10	All	73.57
	62	6	26	All	278.72

Experimental Results: TO

Set	Tasks	PEs	SST Sched. Tasks	TO Sched. Tasks	avg. % increase in δ
1	20	2	10	All	832.88
	26	3	13	All	27.36
	40	4	6	All	88.41
	50	5	10	All	73.57
	62	6	26	All	278.72
2	24	3	5	All	113.95
	25	3	All	All	61.87

Experimental Results: TM

- Benchmarks
  - 7 synthetic
  - 1 real life test case based on the SAE Automotive benchmark

- TM compared to:
  - Straightforward Solution for Messages (SSM)
    - Builds TT schedules with the goal to optimize the end-to-end response time of the TT frames without considering the RC traffic

Experimental Results: TM

Set	Test case	ES	NS	Messages	Frame instances	Δ_{cost} [%]
1	11	13	4	80	12593	2.58
	12	25	6	88	1787	24.44
	13	35	8	103	2285	20.06
	14	45	10	165	3299	11.90

Experimental Results: TM

Set	Test case	ES	NS	Messages	Frame instances	Δ_{cost} [%]
1	11	13	4	80	12593	2.58
	12	25	6	88	1787	24.44
	13	35	8	103	2285	20.06
	14	45	10	165	3299	11.90
2	21	11	4	115	16904	9.17
	22	25	6	179	2523	20.61
	23	35	8	154	3698	39.34

Experimental Results: TM

Set	Test case	ES	NS	Messages	Frame instances	Δ_{cost} [%]
1	11	13	4	80	12593	2.58
	12	25	6	88	1787	24.44
	13	35	8	103	2285	20.06
	14	45	10	165	3299	11.90
2	21	11	4	115	16904	9.17
	22	25	6	179	2523	20.61
	23	35	8	154	3698	39.34
3	automotive	15	3	170	38305	50.88

Conclusions

- Applications of different criticality levels can be integrated onto the same architecture only if there is enough separation:
  - Separation at PE-level achieved with IMA.
  - Separation at network-level using TTEthernet.

- We proposed a Tabu Search based optimization of task mapping and allocation to partitions, and of time partitions.

- Only by optimizing the implementation of the applications, taking into account the particularities of IMA and TTEthernet, are we able to support the designer in obtaining schedulable implementations.