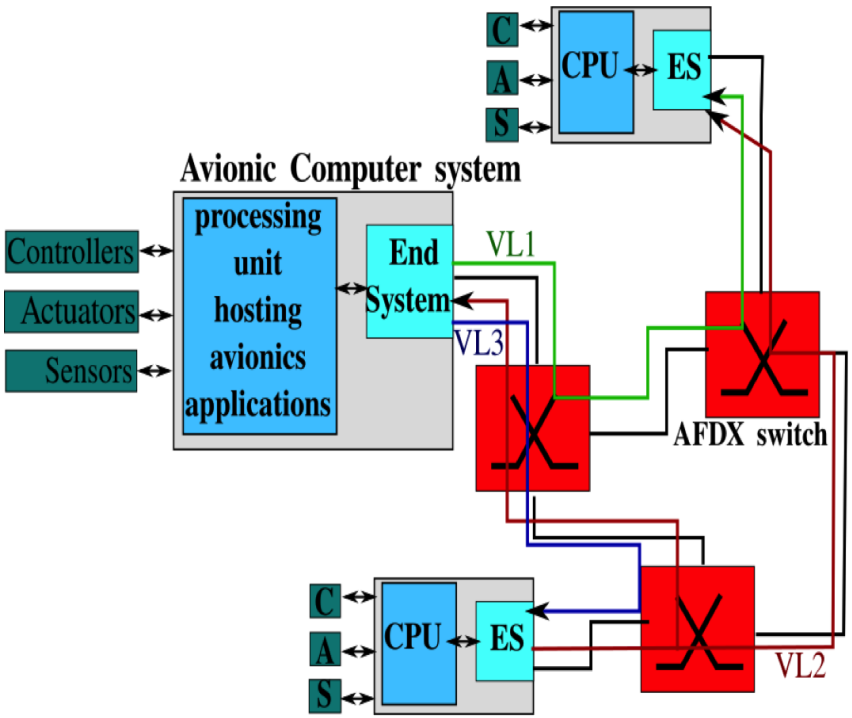
Message scheduling to reduce AFDX jitter in a mixed NoC/AFDX architecture

Jérôme Ermont, Sandrine Mouysset,
Jean-Luc Scharbarg and Christian Fraboul

Université de Toulouse - IRIT - INPT/ENSEEIHT

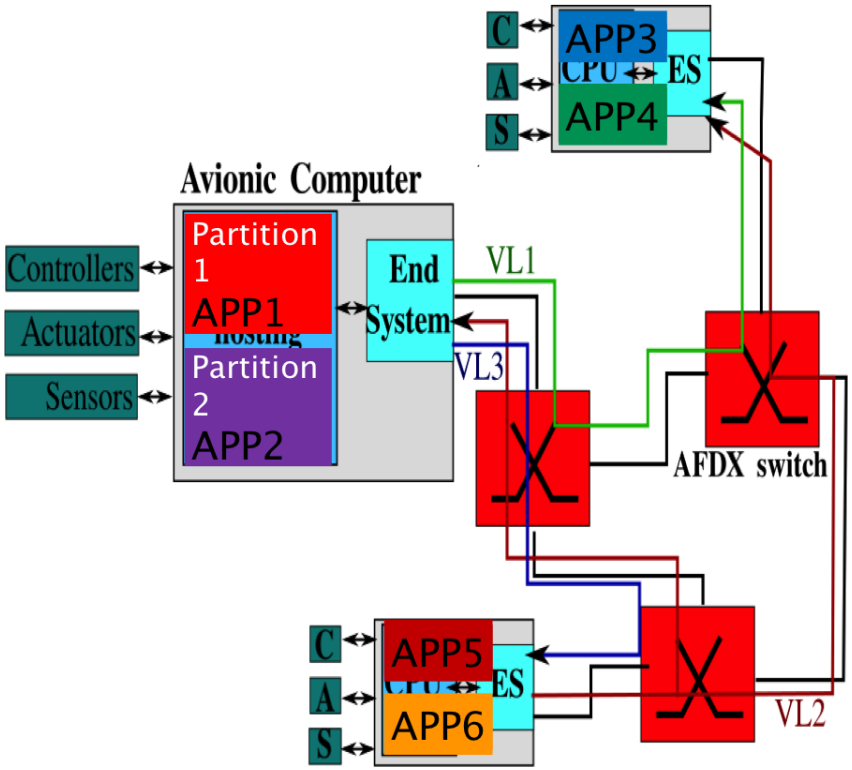
October 12, 2018

Context: Avionics architecture of modern planes



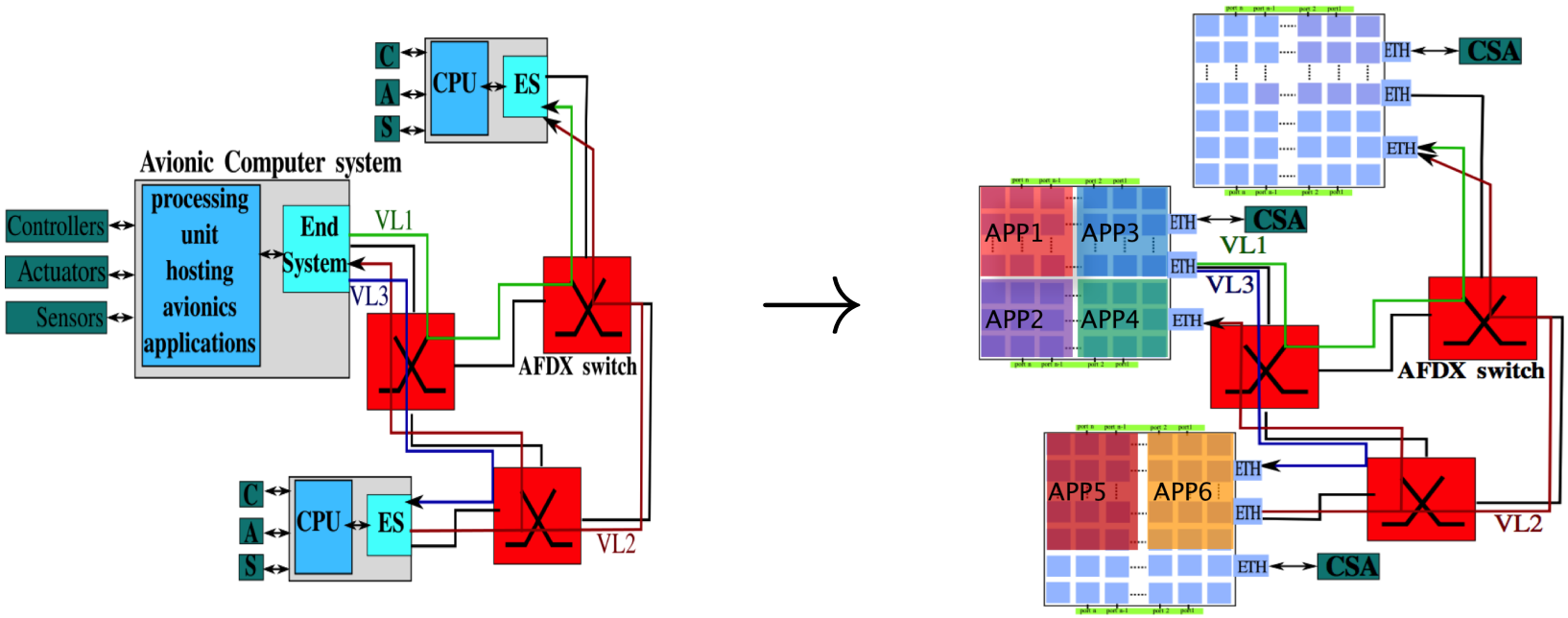
- Avionics computers:
  - Mono-core processors: execute avionics functions following IMA (Integrated Modular Avionics)
  - End Systems: interface between CPU and AFDX
- AFDX network:
  - Interconnection of several avionics computers
  - VL: unidirectionnal flow between one source ES to one or more destination ES
  - BAG: minimum interval time between 2 frames of a VL

Transmission of VLs by an ES



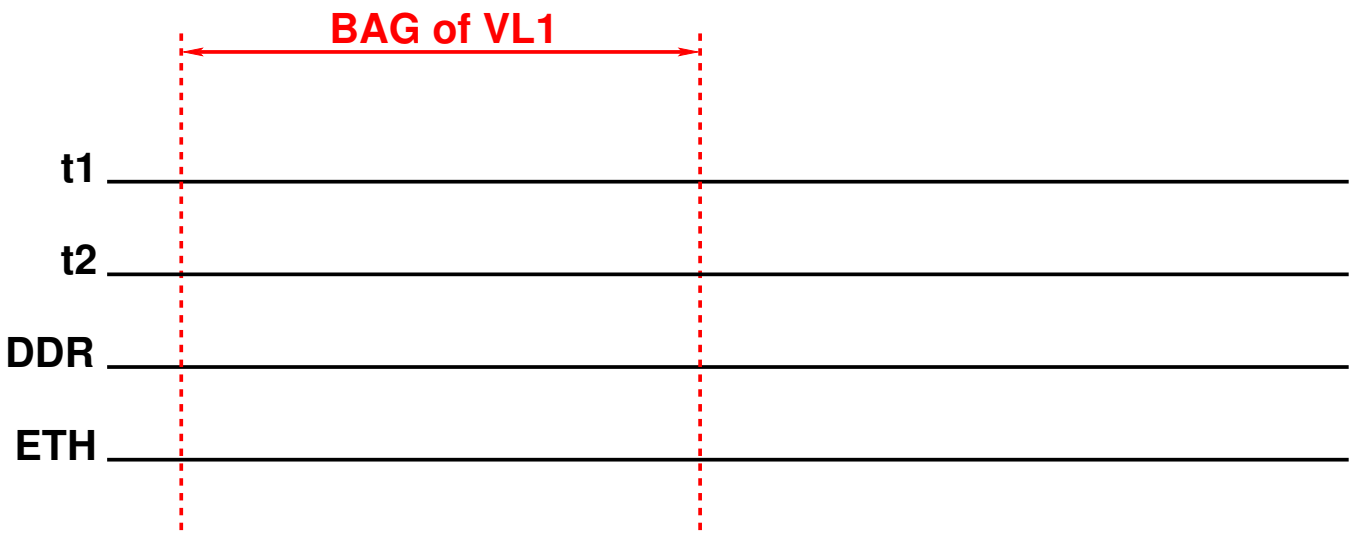
- VLs from different partitions share the same ES
- Scheduler between VLs into the ES
- Introduces a jitter: delay between the beginning of the BAG and the effective transmission of the frame
- AFDX constraint: jitter < 500 μs

Envisioned avionics architecture

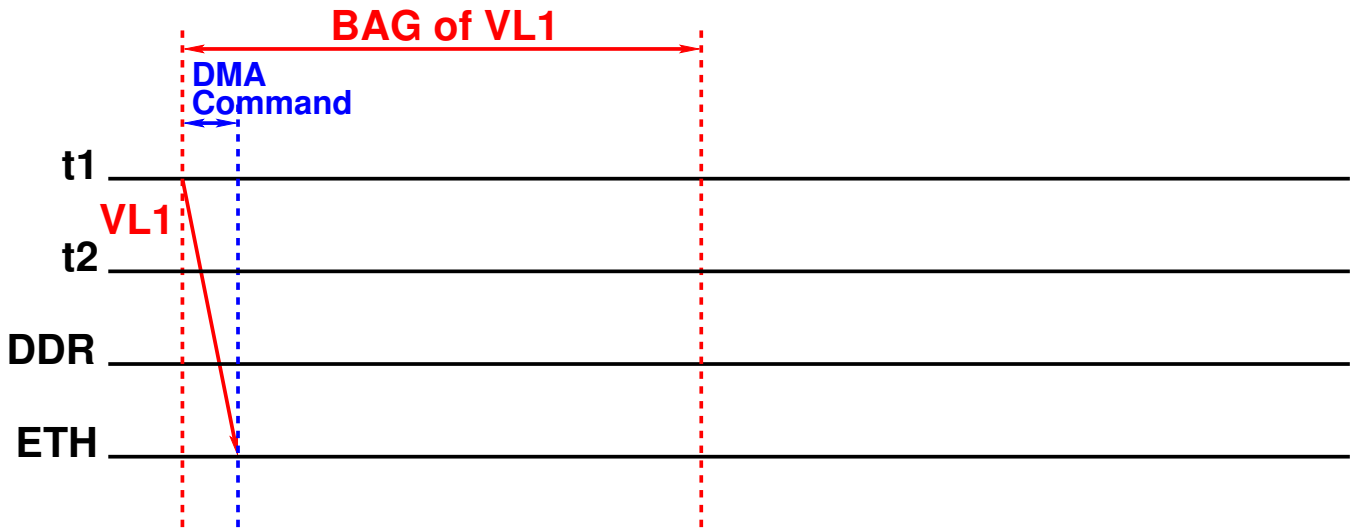


- To replace mono-core processing unit by many-cores
- Different applications can be executed in parallel
- 2 different communications:
  - Intra-NoC communication
  - Inter-NoC communication

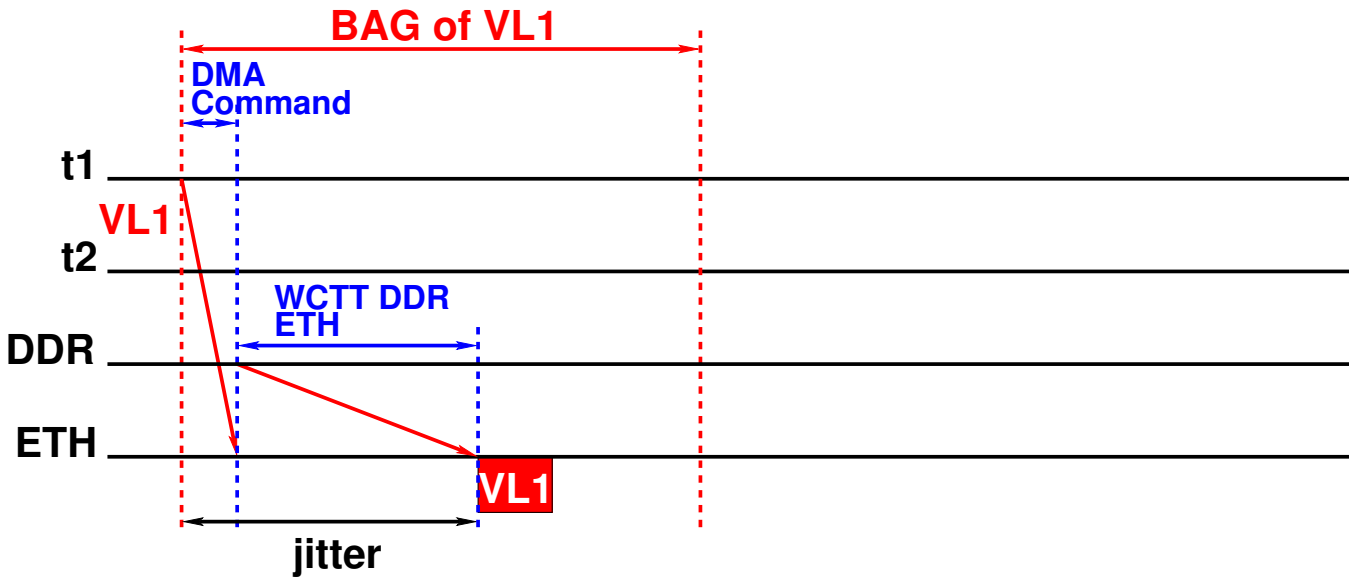
Problem Statement



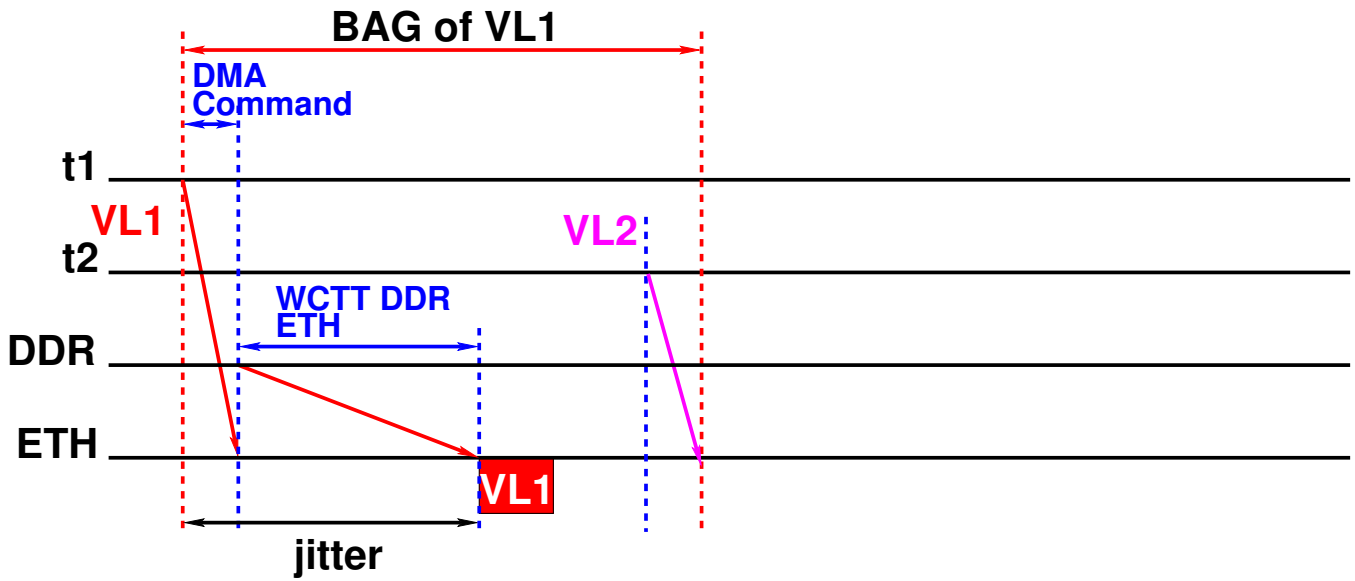
Problem Statement



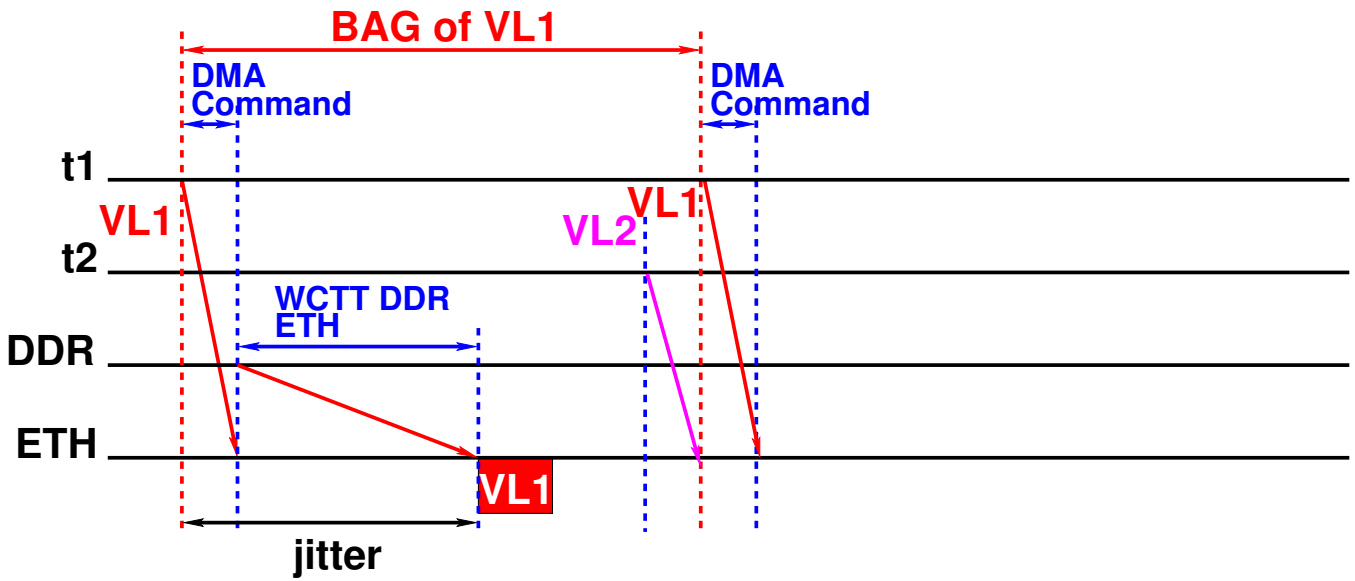
Problem Statement



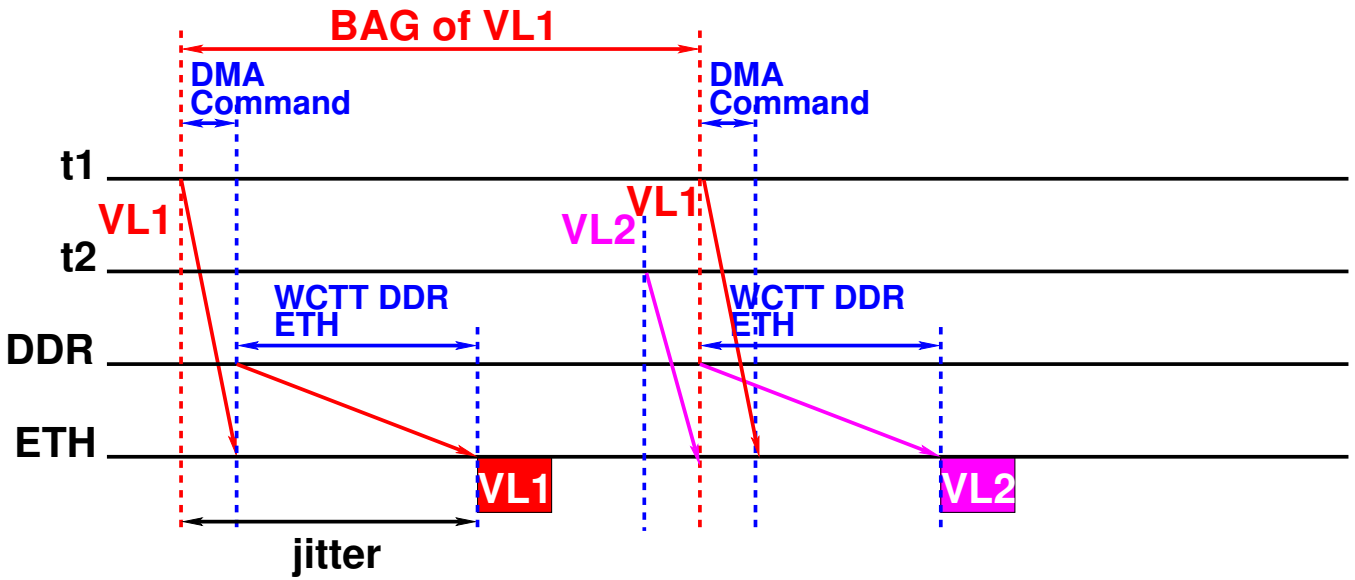
Problem Statement



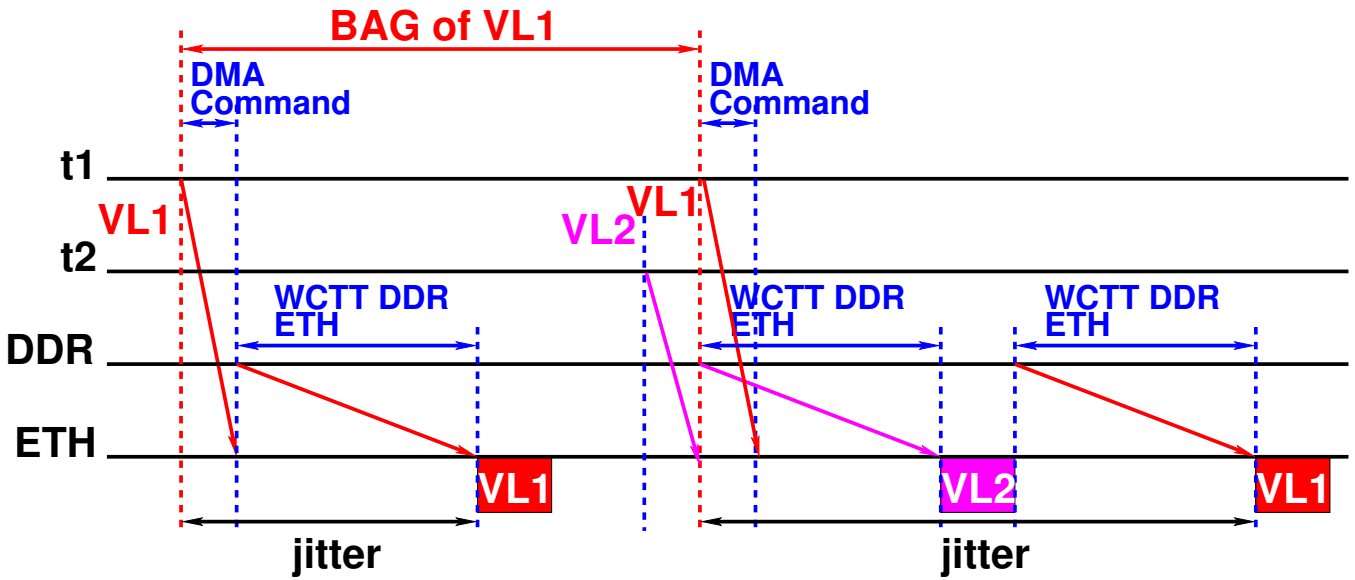
Problem Statement



Problem Statement

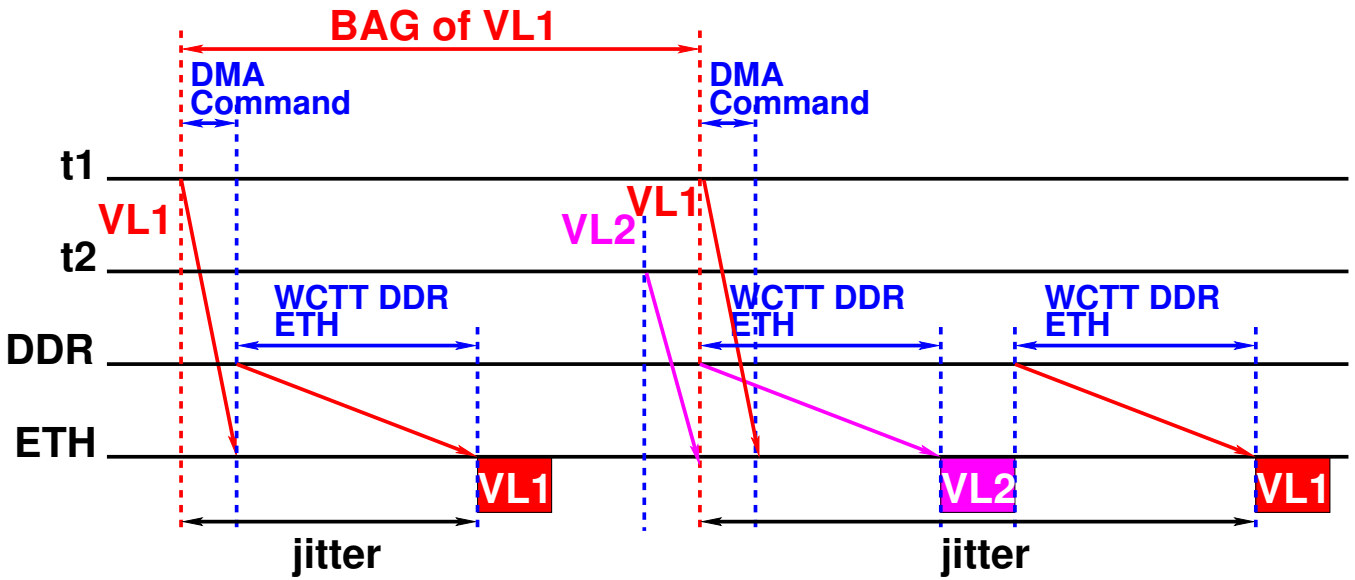


Problem Statement



- The jitter depends on the WCTT of flows from other applications
- WCTT depends on the blocking mechanism of the NoC

Problem Statement



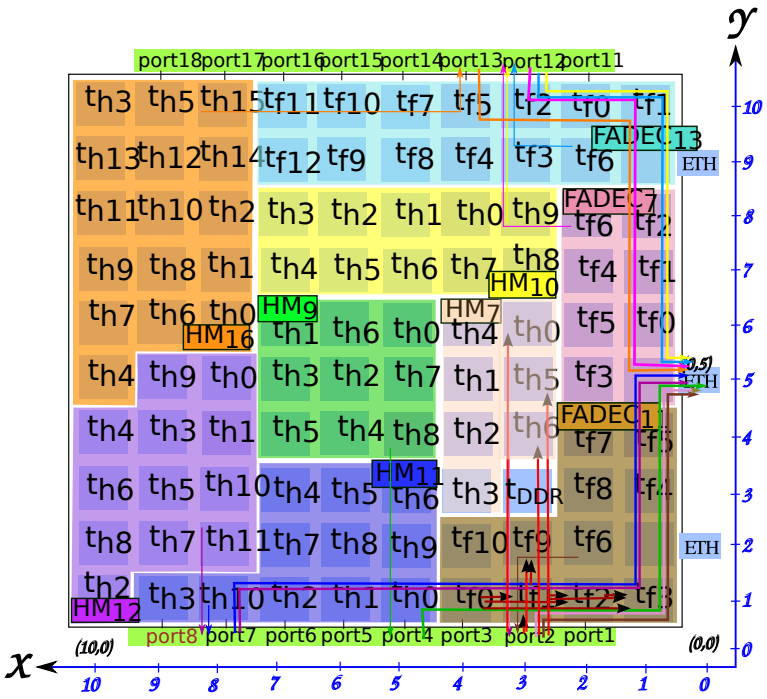
- The jitter depends on the WCTT of flows from other applications
- WCTT depends on the blocking mechanism of the NoC

Problem

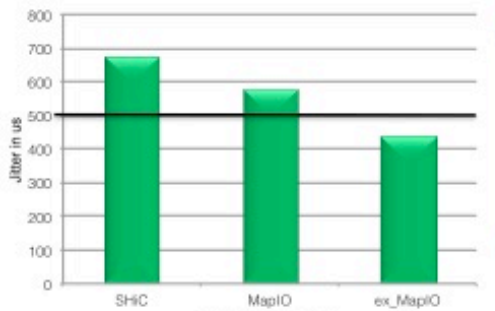
How to reduce the jitter induced by the transmission on the NoC ?

A first solution: minimizing the contention for other applications

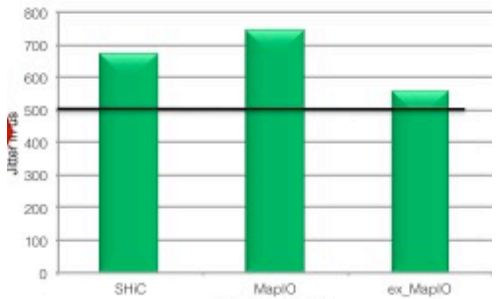
- A new application mapping: Extended Map$_{IO}$ [1]



- Minimizing contentions reduces the maximum jitter



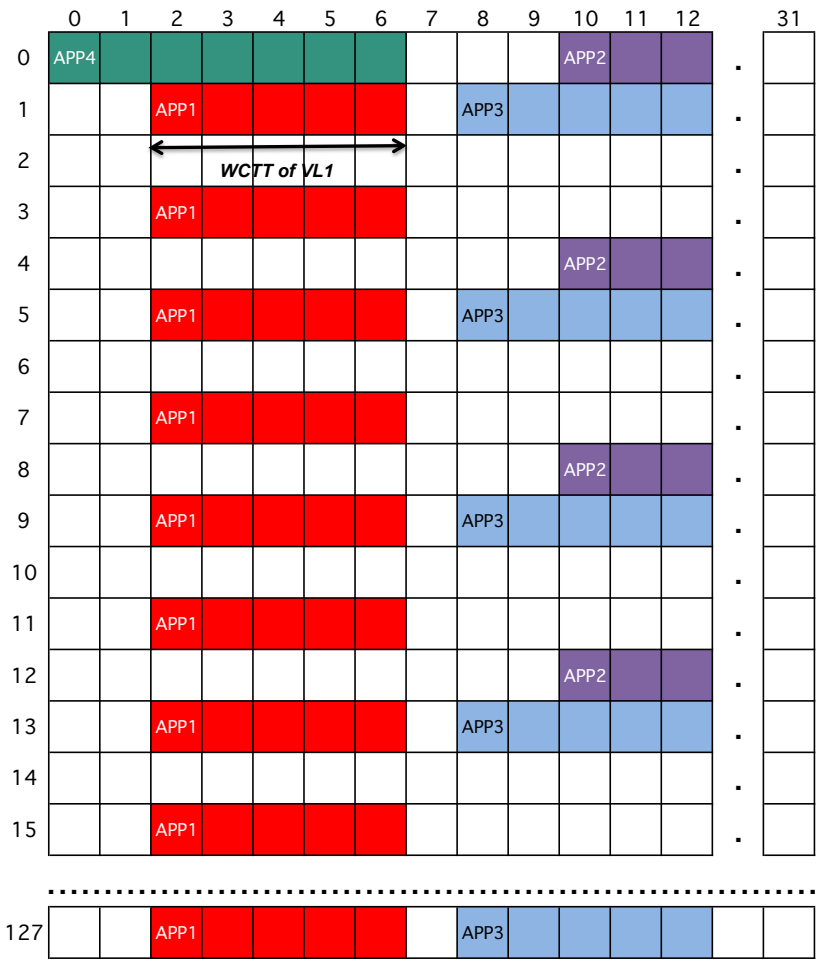
- But not for all configurations

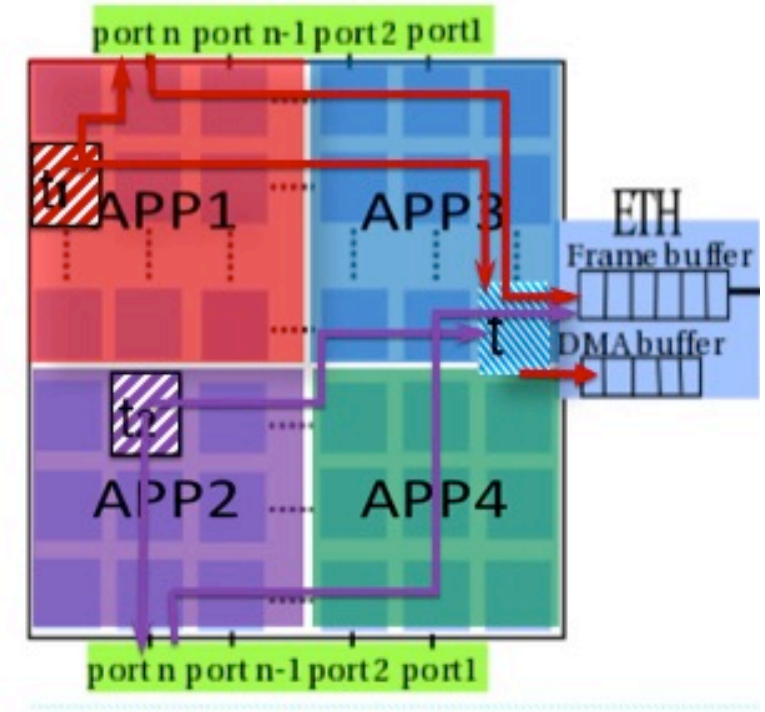


[1] Towards a mixed NoC/AFDX architecture for avionics applications, Laure Abdallah and *al.*, WFCS 2017

Our proposition

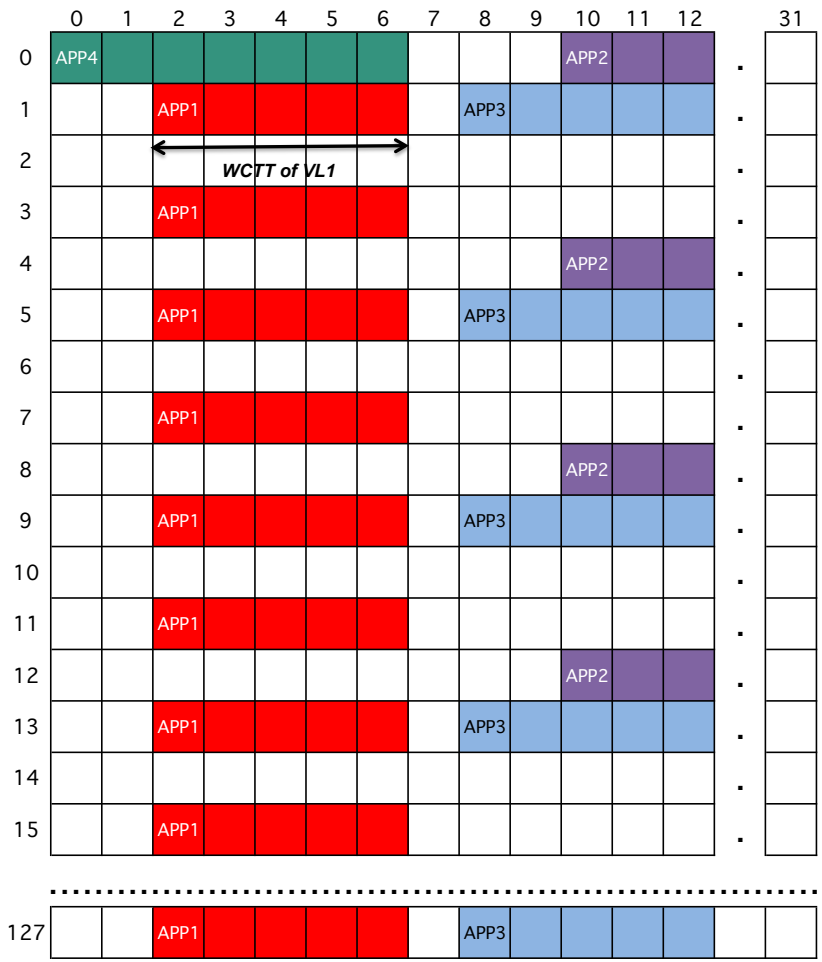
- One node dedicated to schedule the VLs on the ES
- Use of a TDMA table

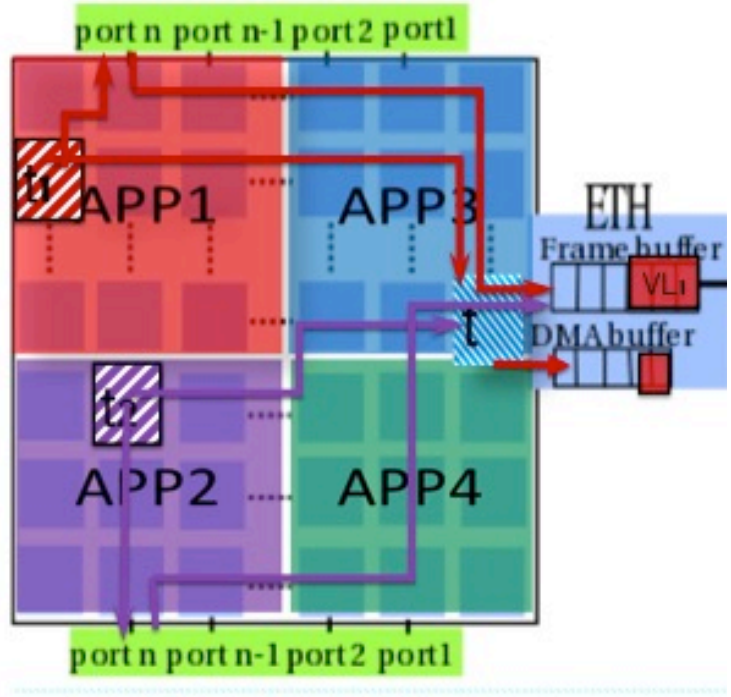




Our proposition

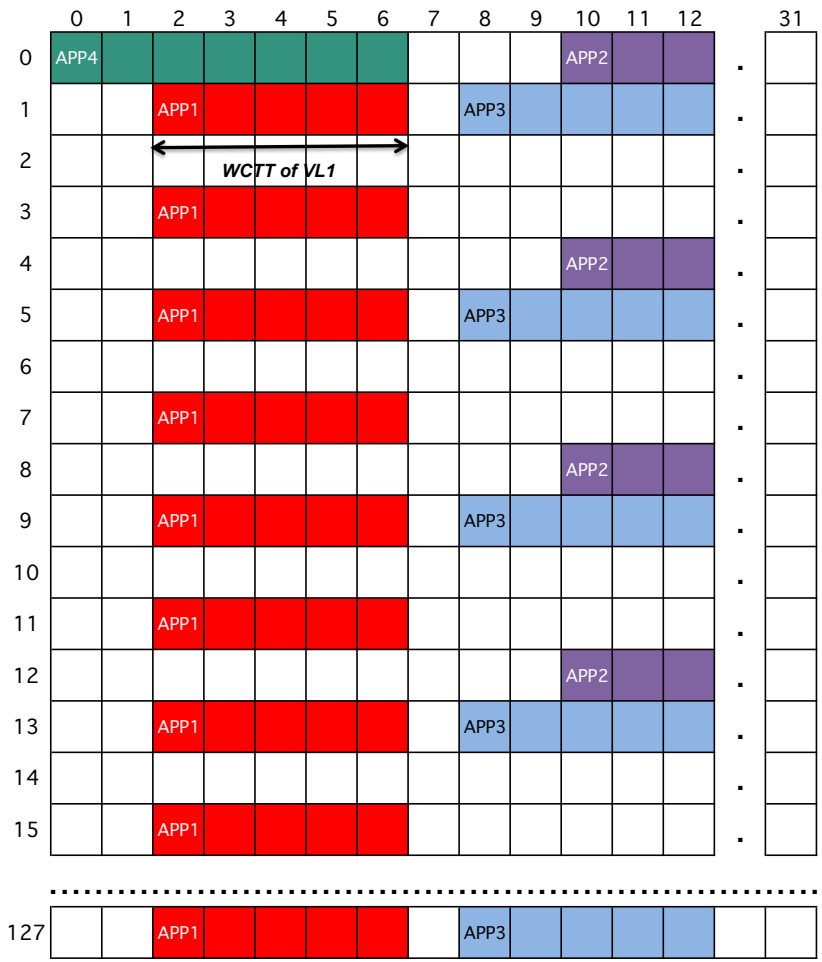
- One node dedicated to schedule the VLs on the ES
- Use of a TDMA table

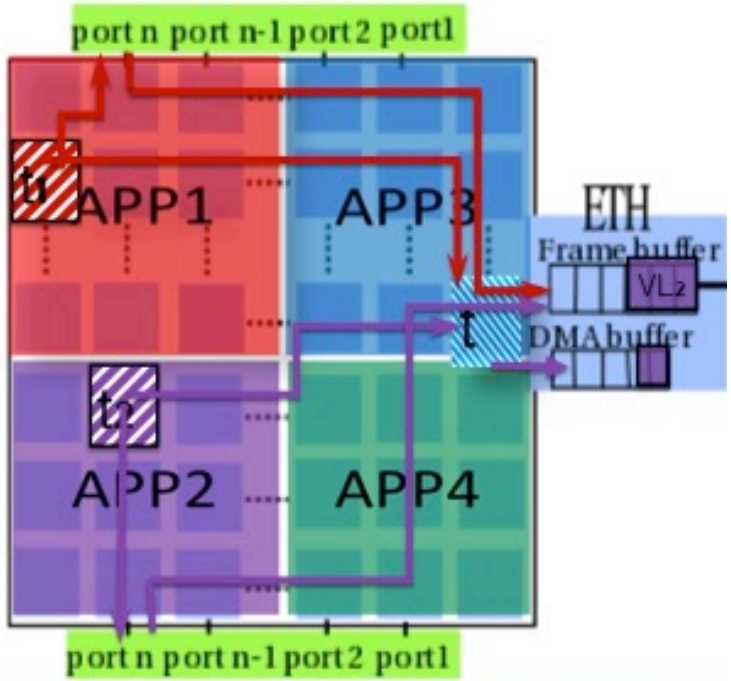




Our proposition

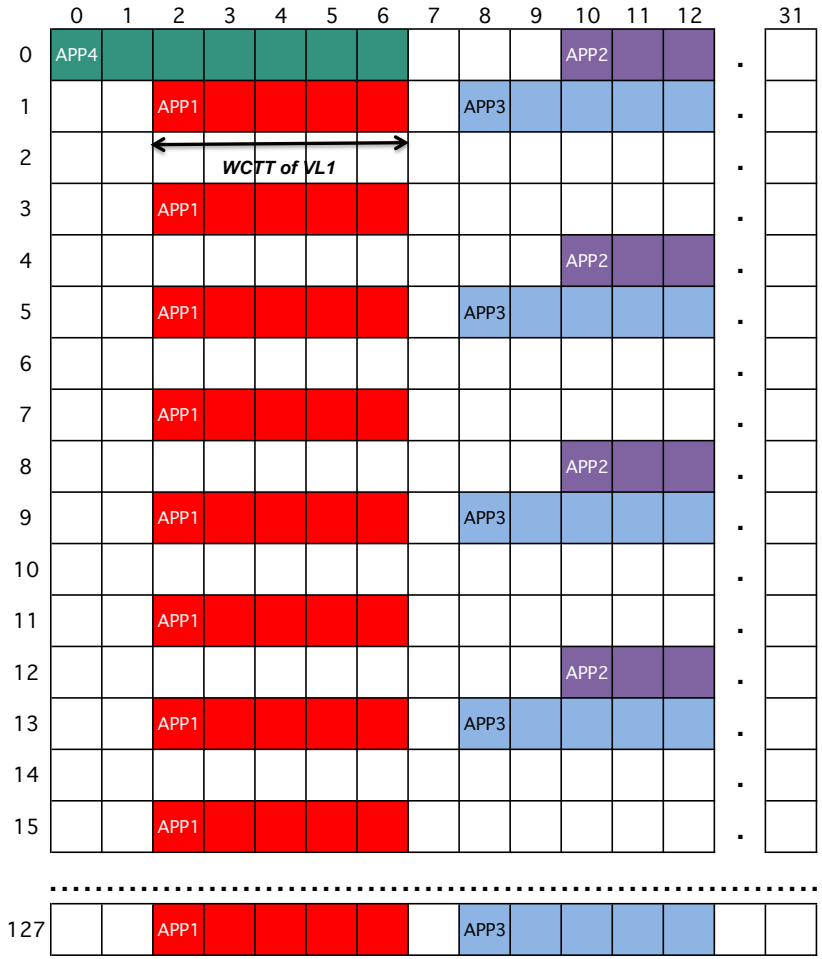
- One node dedicated to schedule the VLs on the ES
- Use of a TDMA table





Our proposition

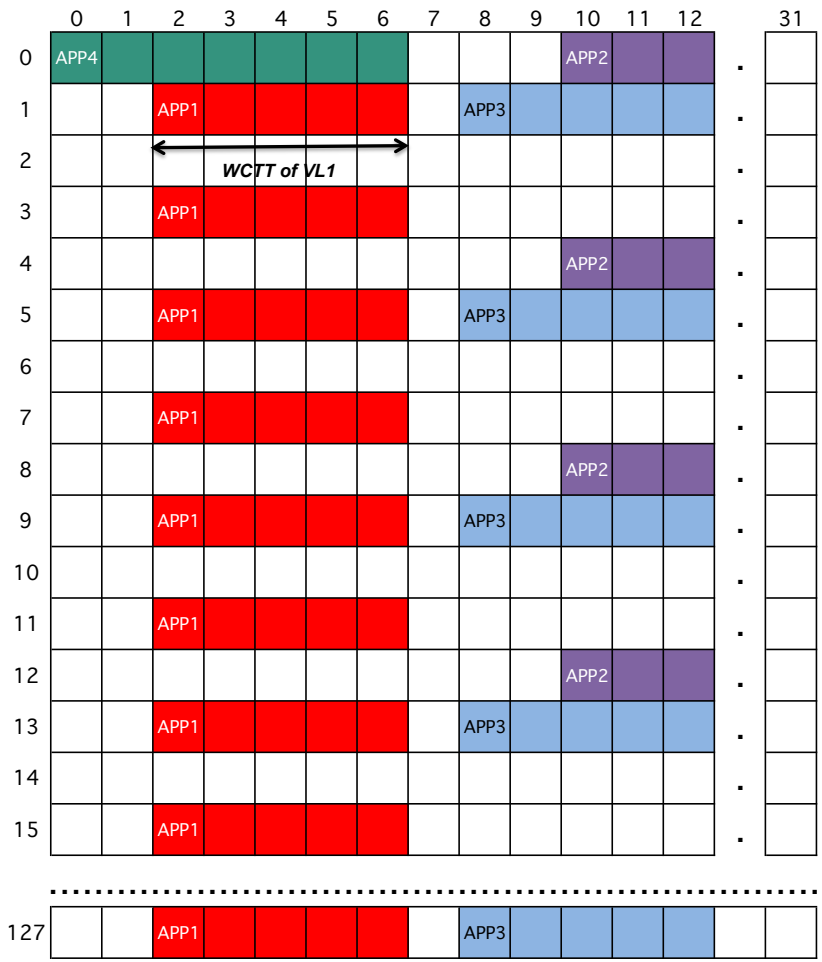
- One node dedicated to schedule the VLs on the ES
- Use of a TDMA table



- Bag of VL4 (from App4): 128 ms
- VL4 is ready when line 6 of the table is executed

Our proposition

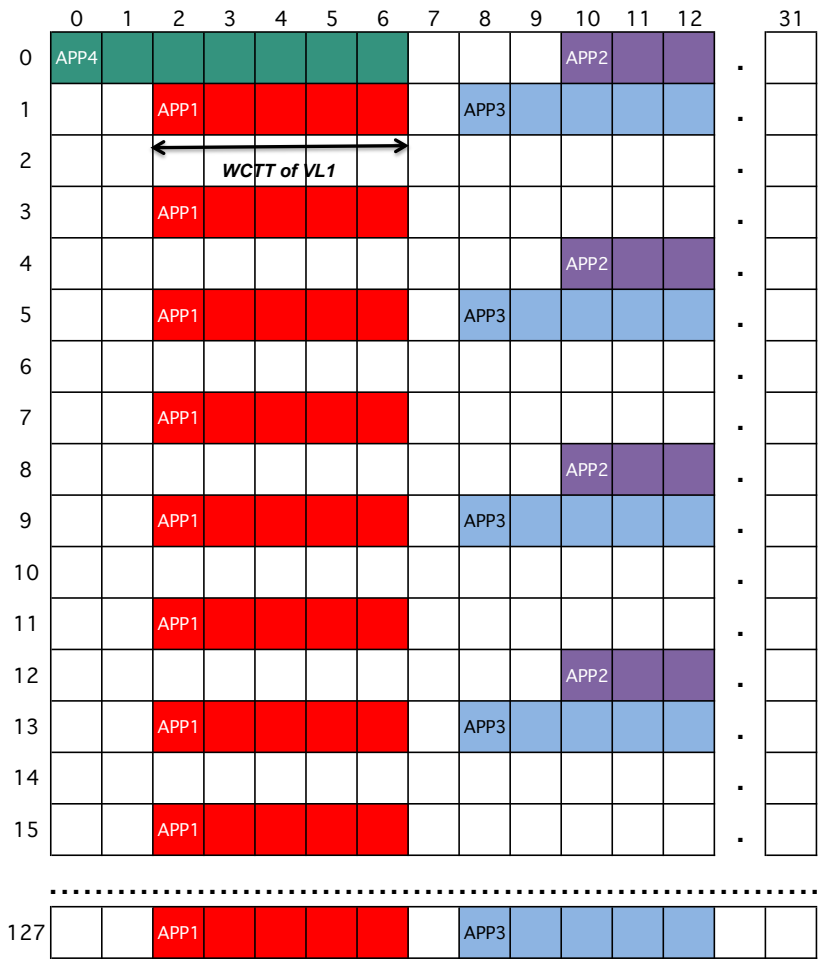
- One node dedicated to schedule the VLs on the ES
- Use of a TDMA table



- Bag of VL4 (from App4): 128 ms
- VL4 is ready when line 6 of the table is executed
- VL4 will wait line 0

Our proposition

- One node dedicated to schedule the VLs on the ES
- Use of a TDMA table



- Bag of VL4 (from App4): 128 ms
- VL4 is ready when line 6 of the table is executed
- VL4 will wait line 0

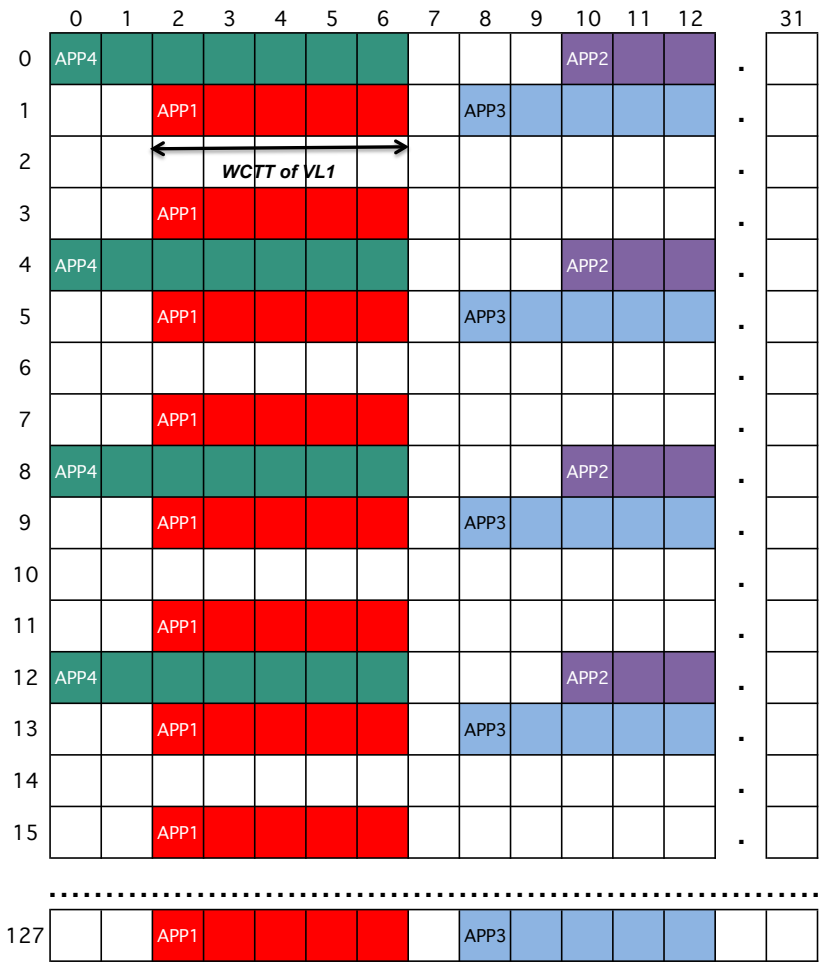
How to reduce this waiting delay ?

Our solution

To give more slots for the VLs
$\rightarrow$ Oversampling of slots

Our proposition

- One node dedicated to schedule the VLs on the ES
- Use of a TDMA table



- Bag of VL4 (from App4): 128 ms
- VL4 is ready when line 6 of the table is executed
- VL4 will wait line 0

How to reduce this waiting delay ?

Our solution

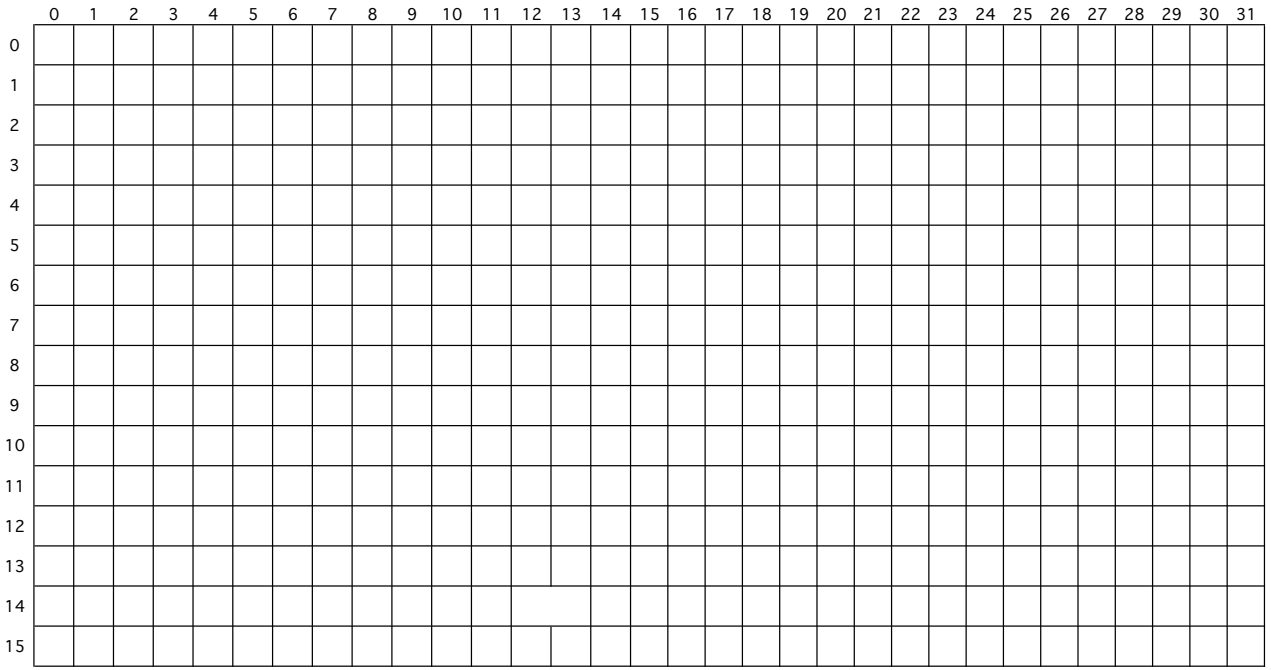
To give more slots for the VLs
$\rightarrow$ Oversampling of slots

How to map the slots to the VLs in the table ?

Constraint

The VLs should respect their BAGs

- VLs with BAG = 1ms allocated to all lines
- Allocation by considering the minimum BAG value ($>$ 1ms)

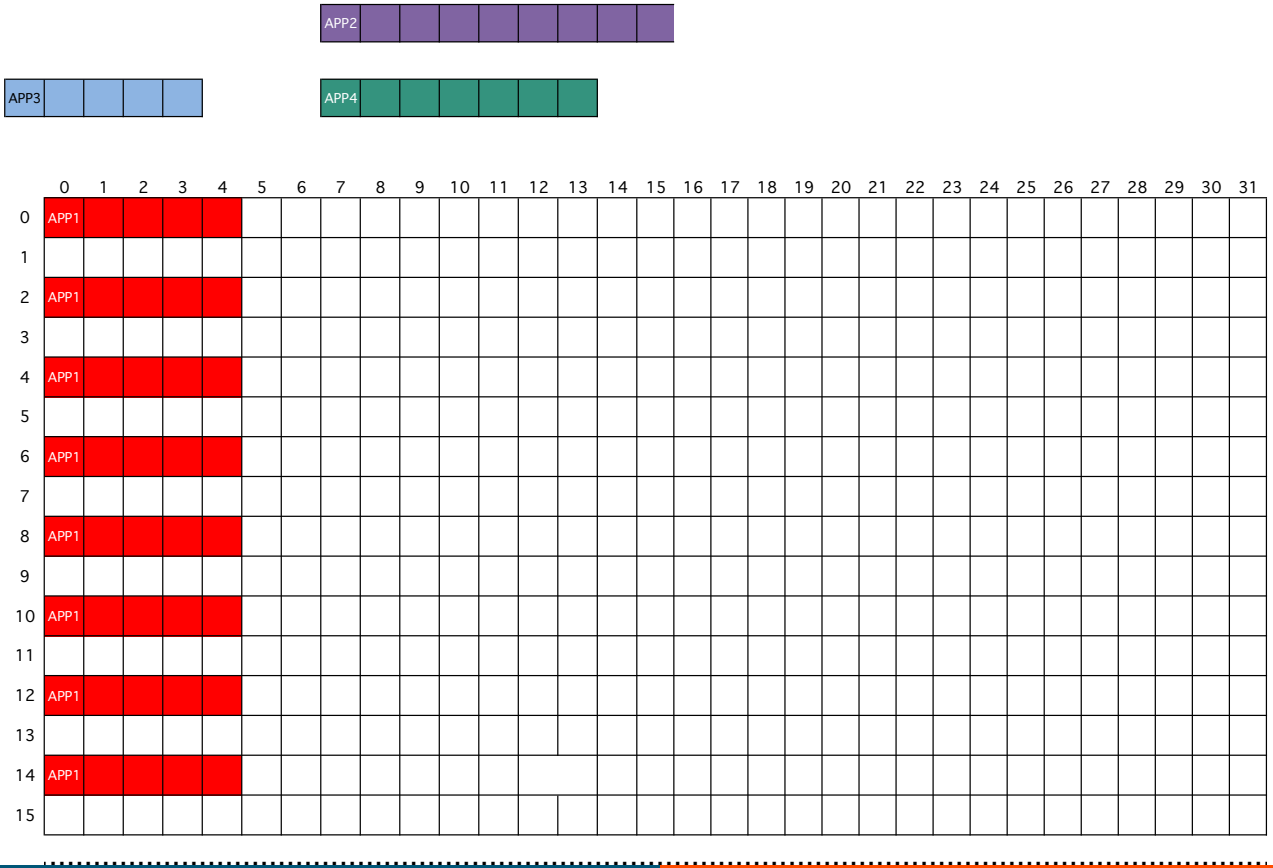


How to map the slots to the VLs in the table ?

Constraint

The VLs should respect their BAGs

- VLs with BAG = 1ms allocated to all lines
- Allocation by considering the minimum BAG value ($>$ 1ms)

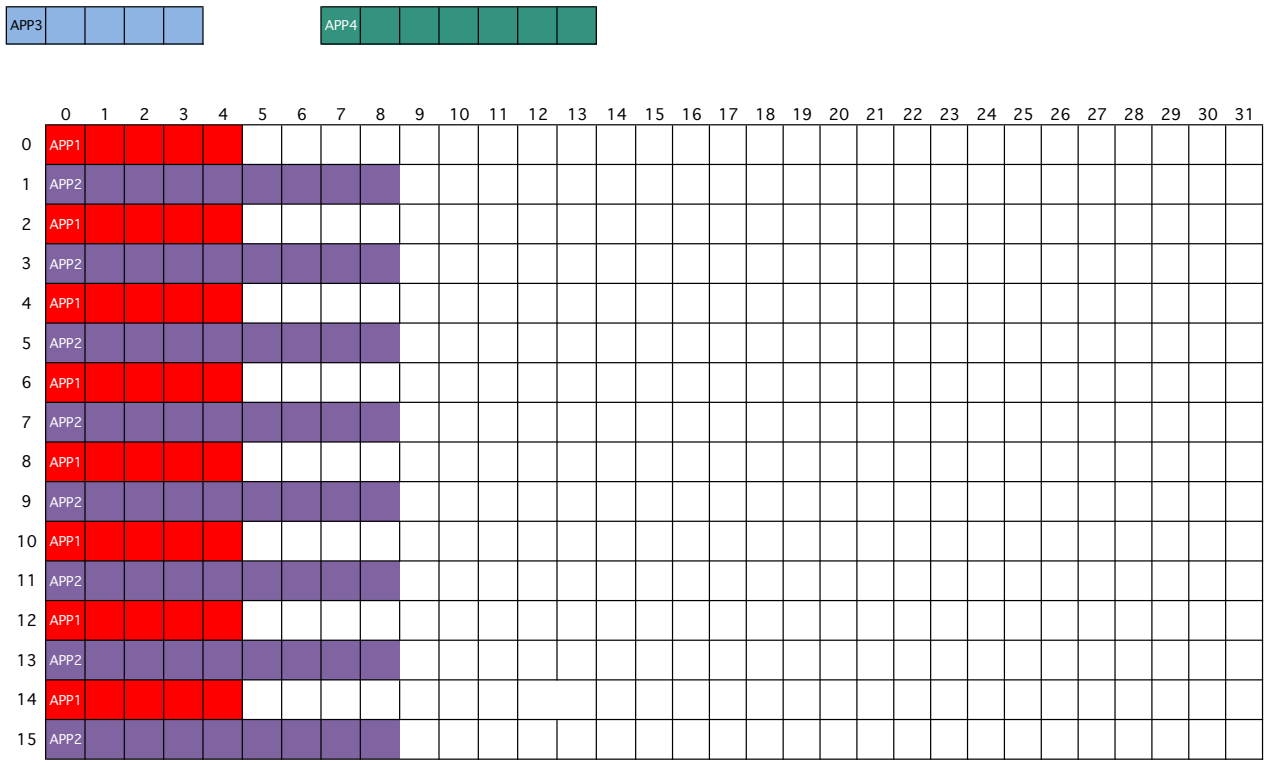


How to map the slots to the VLs in the table ?

Constraint

The VLs should respect their BAGs

- VLs with BAG = 1ms allocated to all lines
- Allocation by considering the minimum BAG value ($>$ 1ms)

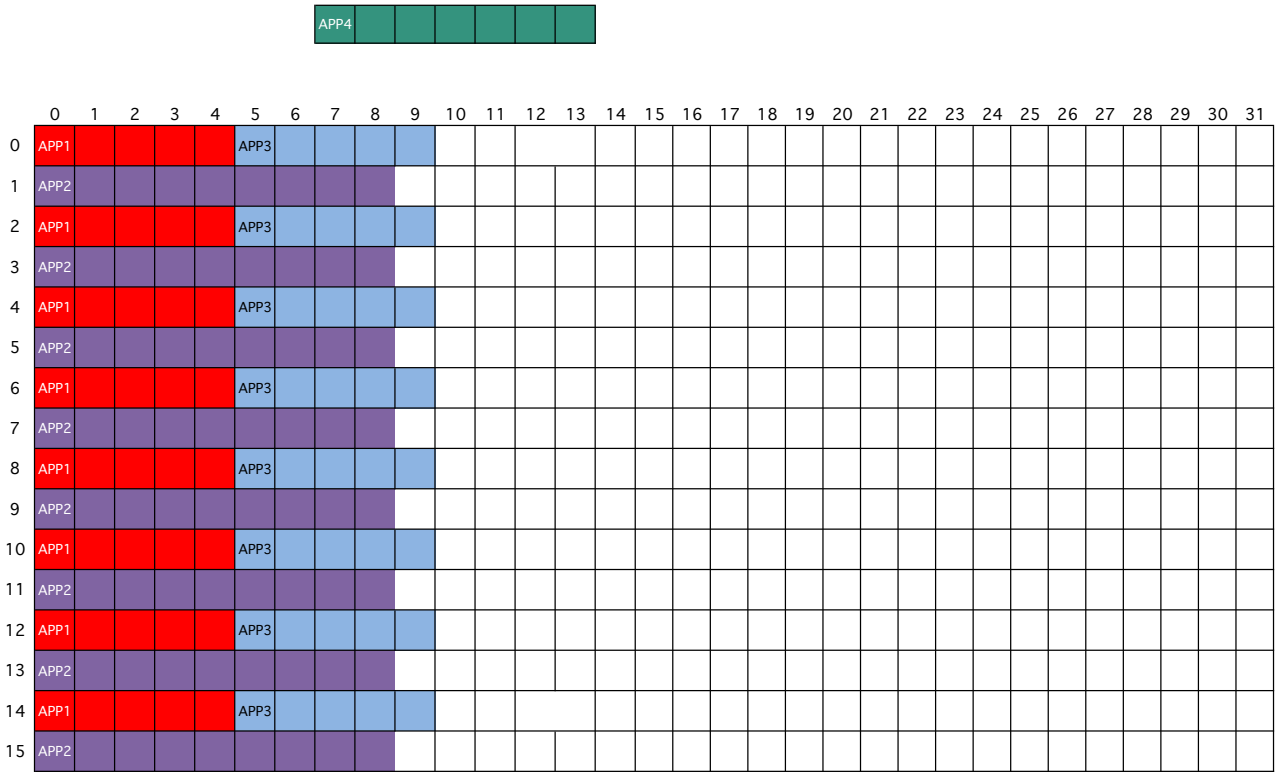


How to map the slots to the VLs in the table ?

Constraint

The VLs should respect their BAGs

- VLs with BAG = 1ms allocated to all lines
- Allocation by considering the minimum BAG value ($>$ 1ms)

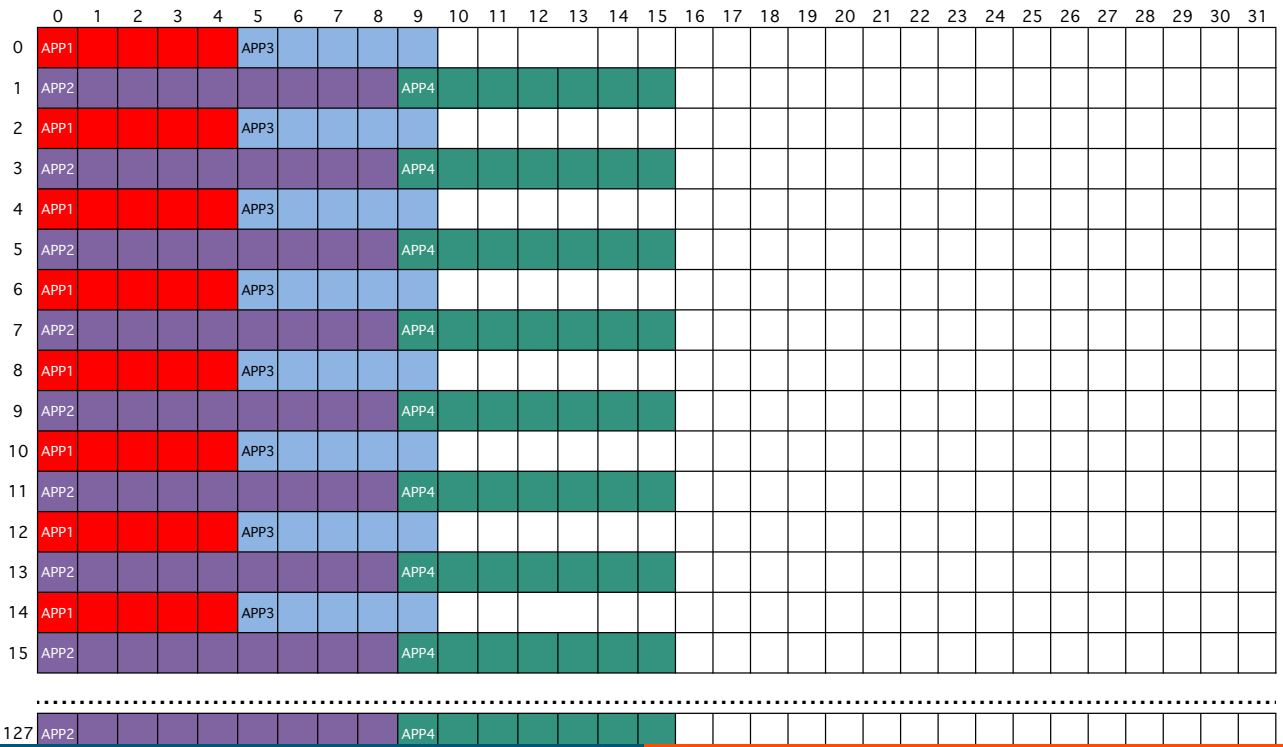


How to map the slots to the VLs in the table ?

Constraint

The VLs should respect their BAGs

- VLs with BAG = 1ms allocated to all lines
- Allocation by considering the minimum BAG value ($>$ 1ms)



Formulation by a Bin Packing Problem

Objective

To allocate VL transmissions into a minimum number of lines

- Number of lines in which VLs are allocated

$$N = \min_{j=1\ldots m,\ BAG_j \neq 1} BAG_j$$

- Objective function

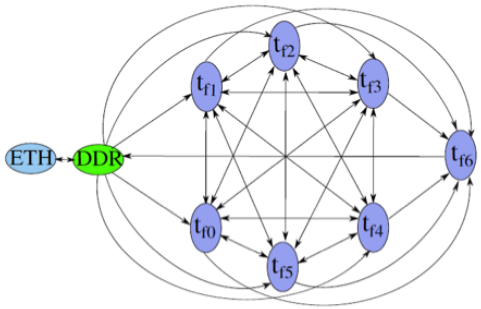
$$\begin{aligned}
\min \quad & \sum_{i=1}^{N} y_i & \\
s.t \quad & \sum_{j=1}^{m} \omega_j x_{ij} & \leq C\ y_i, \forall i = 1, .., N \\
& \sum_{i=1}^{N} x_{ij} & = 1, \forall j = 1, .., m \\
& y_i \in \{0, 1\} & , \forall i = 1, .., N \\
& x_{ij} \in \{0, 1\} & , \forall i = 1, .., N, \forall j = 1, .., m.
\end{aligned}$$

Evaluation case study

- A 10x10 Tilera-like NoC
- 2 types of applications:

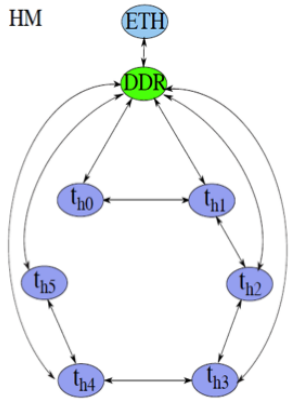
FADEC: engine control

- critical application
- little amount of exchanged data: 1500 bytes
- full transmission between task



HM: HouseKeeping

- non-critical application
- lots of data exchanged: $>$ 130 Koctets
- data are stored in the memory



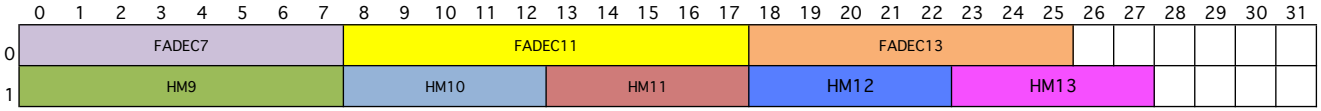
Evaluation case study

- 9 (critical or non-critical) considered applications: $FADEC_7$ (4), $FADEC_{11}$ (8), $FADEC_{13}$ (16), HM_7 (4), HM_9 (2), HM_{10} (16), HM_{11} (32), HM_{12} (16), HM_{16} (32)
- 2 system configurations:
  - 8 applications: HM_7 is removed
  - 9 applications
- 3 mapping strategies:
  - SHiC [1]: mapping by considering the core-to-core communications
  - Map_{IO} [2]: mapping by considering core-to-IO communications
  - $exMap_{IO}$ [3]: mapping by considering both core-to-IO and IO-to-core communications

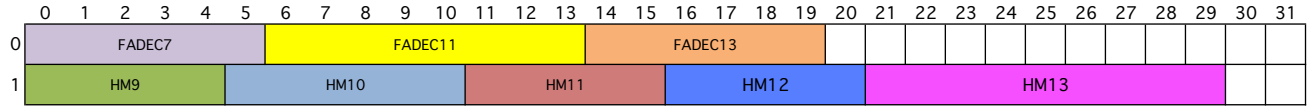
[1] Smart hill climbing for agile dynamic mapping in many-core systems, Mohammad Fattah and *al*
[2] Reducing the contention experienced by real-time core-to-I/O flows over a Tilera-like Network on Chip, Laure Abdallah and *al.*, ECRTS 2016
[3] Towards a mixed NoC/AFDX architecture for avionics applications, Laure Abdallah and *al.*, WFCS 2017

VLs transmissions packed into the table

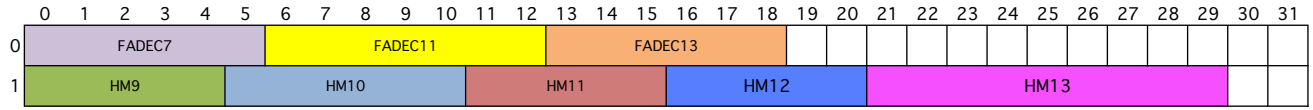
- SHiC



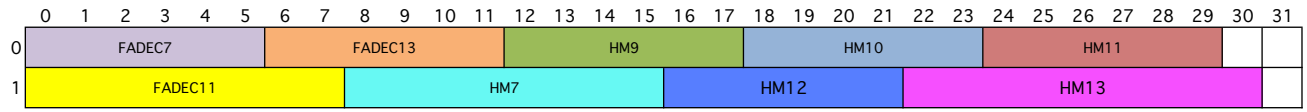
- Map_{IO} with 8 applications



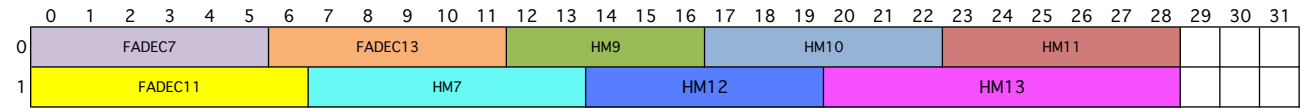
- Ex_Map_{IO} with 8 applications



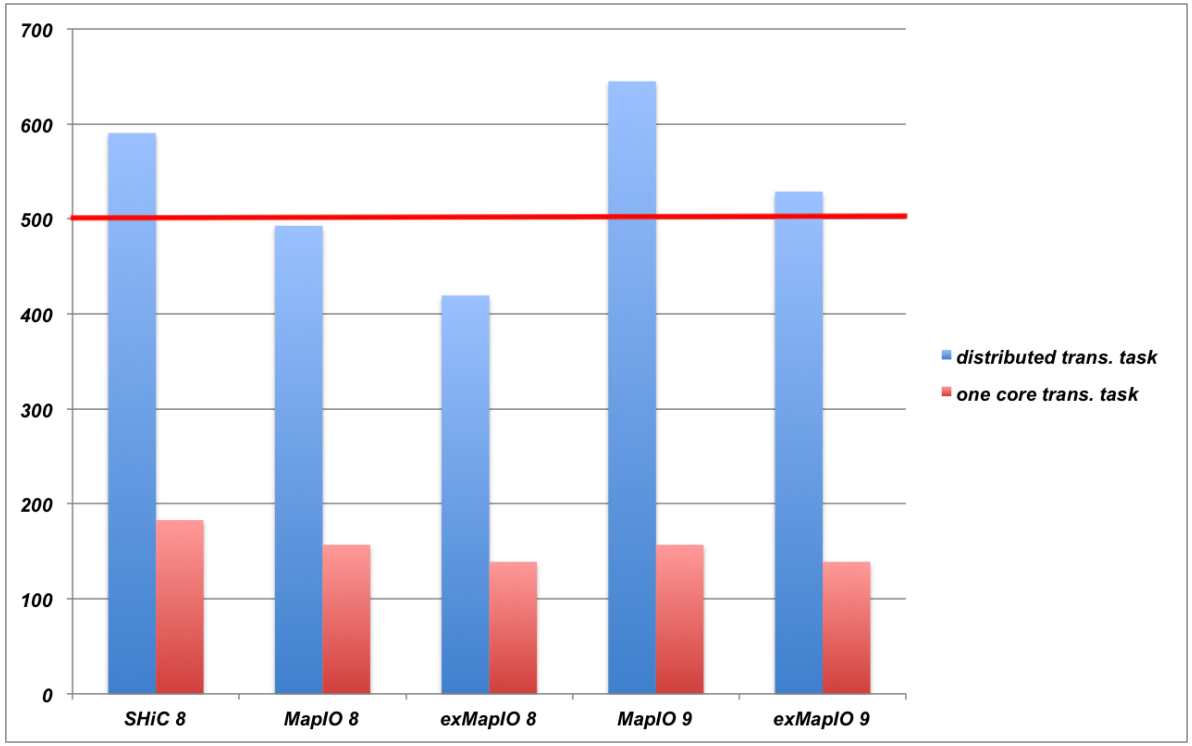
- Map_{IO} with 9 applications



- Ex_Map_{IO} with 9 applications



Results



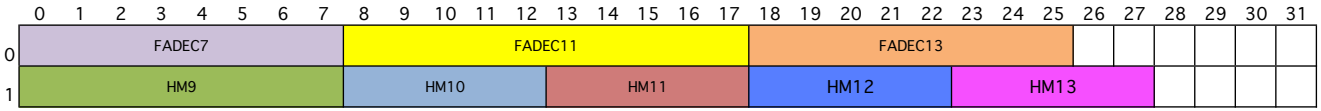
- TDMA table guarantees the transmission every BAG
- Jitter constraint is respected when using a dedicated node

Conclusion

- To replace mono-core processors by NoC based many-cores architecture
- Sharing the same output port could lead to an execution for which the jitter constraint is exceeded
- Mapping strategy Extended Map_{IO} minimizes the jitter by reducing the contention
  - But jitter constraint can be exceeded
- Our proposition: one dedicated node schedules the outgoing flows using a TDMA table
  - The jitter only depends on the contentions for the outgoing flow
  - The jitter is then significantly reduced
- Construction of a scheduling table
  - Guarantee of the BAG constraint
  - Over allocation of slots in order to reduce waiting delays

Further works

- SHiC mapping example



  - What happens if HM_{13} needs 10 slots ?
  - Different possible solutions
    - Reduce more the contentions on outgoing flows
    - Relax the constraint of the minimum number of lines for larger BAG value $\rightarrow$ Variable capacity size bin packing or cutting stock problem
- Global transmission delay from one manycore to another via AFDX
- Implementation of the solution in a real manycore system such as Tilera or Kalray