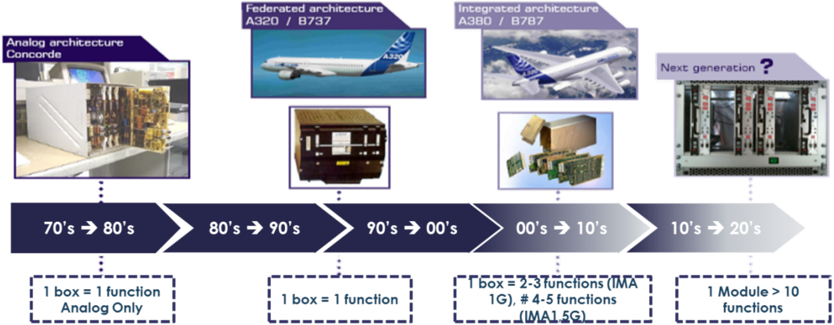
Towards a NoC/AFDX avionics architecture with heterogeneous flows

Jean-Luc Scharbarg
Université de Toulouse - IRIT/ENSEEIHT/INPT
Jean-Luc.Scharbarg@enseeiht.fr

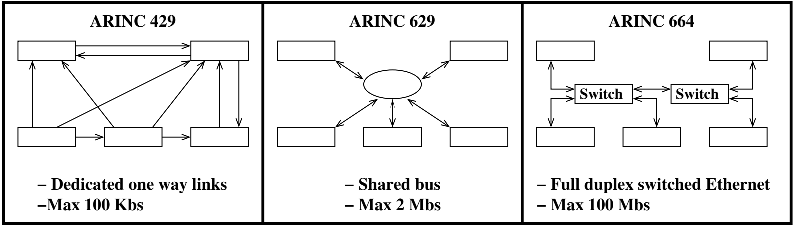
ETR 2017 - Paris

Avionics architecture evolution



- 1 module = 1 manycore ⇒ mapping of avionics functions on these distributed manycore architectures

Avionics communication evolution



- Under utilisation of the network ⇒ QoS facilities to leverage spare bandwidth for the transmission of non avionic flows

Summary

1. From classical avionics to IMA
2. Distributed manycore architectures
3. Avionics and non avionics flows on the same network
4. Conclusion

Summary

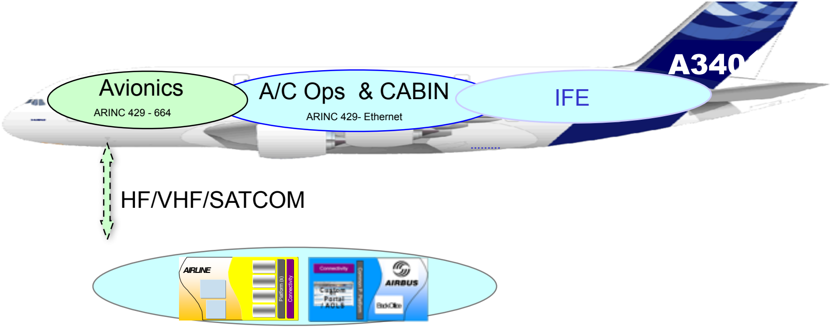
1. From classical avionics to IMA

2. Distributed manycore architectures

3. Avionics and non avionics flows on the same network

4. Conclusion

Civilian aircraft communications



Avionics

ARINC 429 - 664

A/C Ops & CABIN

ARINC 429- Ethernet

IFE

HF/VHF/SATCOM

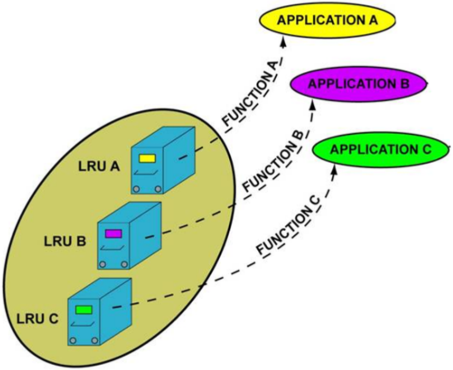
Ground Information System (Airlines, Airbus,...)

Avionics systems characteristics

- Avionics: electronic systems installed in an aircraft
  - Calculators and their software, sensors and actuators
  - Communication links between these elements
- Example of avionic systems: autopilot, navigation, flight control, . . .
- Constraints on the avionics systems
  - Volume and weight limitations
  - Correct behavior in severe conditions: heat, vibrations, electromagnetic interferences, . . .
  - Safety level required, depending on the system (ARP 4754): catastrophic (failure = loss of the aircraft), hazardous, major, minor, no effect
  - Segregation between critical functions
- Aeronautical Radio INCorporated (ARINC): leadership in the development of specifications and standards for avionics equipment
- Airlines Electronic Engineering Committee (AEEC): development and maintenance of specifications and standards (airlines, governments, ARINC)
- ADN: Aircraft Data Network

Classical avionics architecture (up to A340)

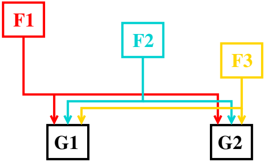
- Each equipment (LRU) dedicated to a system function (braking, flight computers, . . .): sensors, actuators, calculators . . .
- natural segregation between the equipments



CONVENTIONAL AVIONICS
"Line Replaceable Unit"
(LRU)

Data transmission in classical avionics

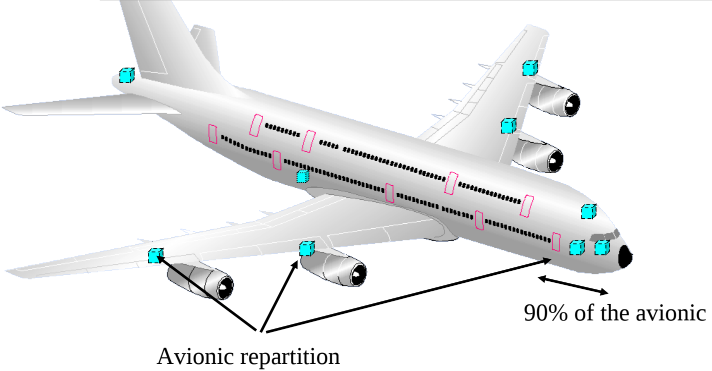
- A network of equipments



- Each data is transmitted from its source to every equipment that needs the data
- One dedicated line for each data
- Repetitive transmission of the data on each line
- Each line: a mono-emitter ARINC 429 data bus
- Each data individually identified by a label
- Low bit rate: 12 Kbits/s to 100 Kbits/s
- At most 20 receivers per line

Classic avionics evolution

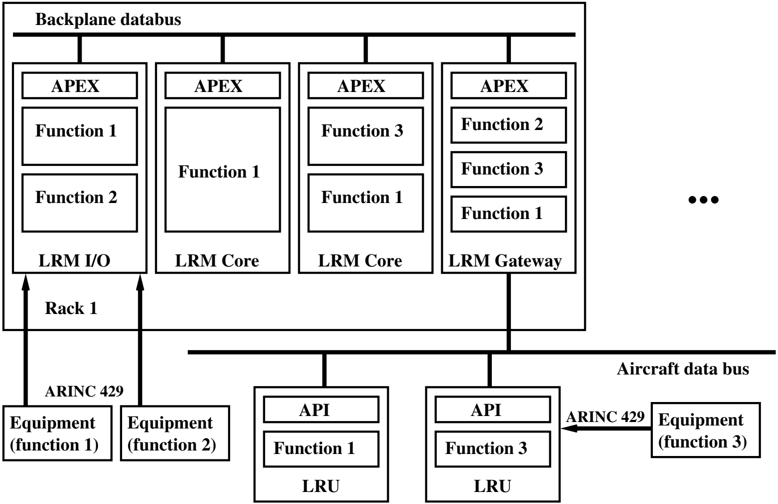
	A310 (1983)	A320 (1988)	A340 (1993)	A380 (2005)
Avionic Volume	745 litres	760 litres	830 litres	
Number of equipment	77	102	115	147
Embedding Software	4 Mo	10 Mo	20 Mo	100 Mo
Number of ARINC 429	136	253	368	1000 (AFDX)
Computing Power	60 Mips	160 Mips	250 Mips	



90% of the avionic

Avionic repartition

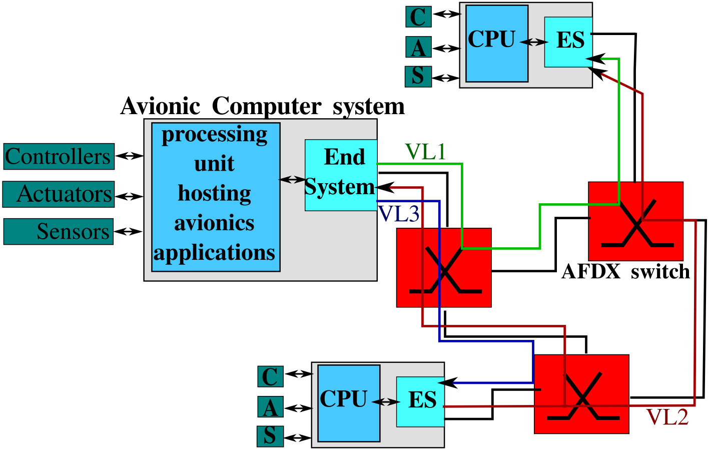
IMA architecture (ARINC 651)



IMA characteristics

- Hardware components
  - LRM (Line Replaceable modules): resources in common cabinets
    - Core modules for the execution of the applications
    - Input/output modules for communications with non-IMA equipments
    - Gateway modules for communications between cabinets
  - LRU (Line Replaceable Unit): existing non-IMA equipments
  - Backplane databus: ARINC 659
- Sharing of execution resources
  - An avionics subsystem: a partition with an assigned time window to execute its application on a shared module
  - Guarantee isolation between subsystems ⇒ robust partitioning concept
    - Spatial isolation: limitation and protection of the address space of each partition, e.g. by a MMU
    - Temporal isolation: static allocation of a time slice for the execution of each partition, based on WCET
  - Communications between partitions via ports (APEX: APplication EXecutive)
    - Sampling ports: only the last value of data is stored
    - Queueing ports: all the values of data are stored
  - Logical channel: multicast link between ports, independent of the communication technology

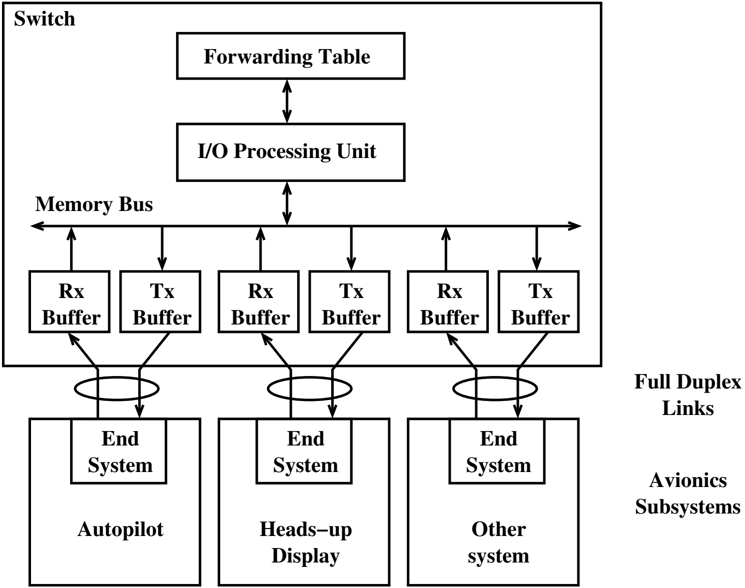
Today avionics architecture



The reasons for the AFDX

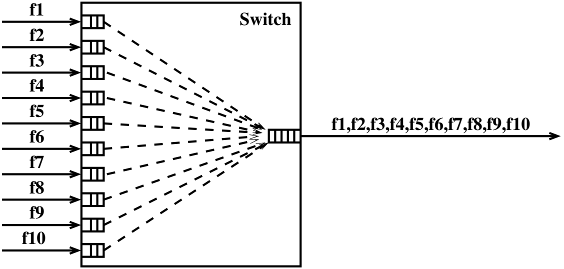
- ARINC 429 is no more sufficient: too many buses are needed
- ARINC 629 is too expensive
- Why Ethernet technology ?
    - High throughput offered to the connected units (100 Mbits/s)
    - High connectivity offered by the network structure
    - A mature industrial standard
    - Low connection cost
- But Ethernet CSMA/CD is not deterministic
    - Potential collision on the physical medium
    - Binary Exponential Back off retransmission algorithm
- The solution adopted for ARINC 664
    - switched Ethernet technology: units directly connected by point-to-point links to Ethernet switches ⇒ possible collision domain = single link between two elements
    - Full duplex links ⇒ no more collisions
- The problem is shifted to the switch level

Full duplex switched Ethernet example



Full duplex switched Ethernet is not deterministic

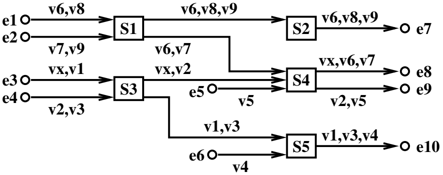
- Temporary congestion on an output port
  - Increase of the waiting delay of frames in the Tx buffer
  - Frame loss by overflow of the Tx buffer
- An illustrative example



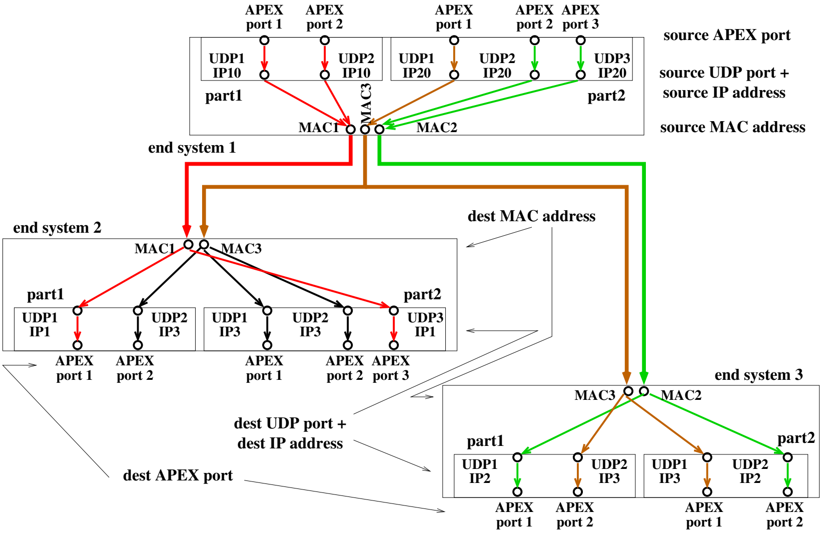
  - Five frames at the same time ⇒ one frame waits until the transmission of the four other ones
  - More than five frames at the same time ⇒ at least one frame is lost
- Addition of dedicated mechanisms to classical full duplex switched Ethernet in order to guarantee the determinism of an AFDX network

The AFDX key characteristics

- Static full duplex Switched Ethernet: no CSMA/CD, no spanning tree, static 802.1D tables
- One FIFO buffer per output port
- Traffic characterization based on the Virtual Link (VL) concept
  - Static definition of the flows which enter the network
  - Multicast path with deterministic routing
  - Mono transmitter assumption
  - Sporadic flows (*BAG*: Bandwidth Allocation Gap)
    - ★ Discrete values: 1, 2, 4, 8, 16, 32, 64, 128 ms
    - ★ Traffic shaping on each emitting end system
  - Minimum (S_{min}) and maximum (S_{max}) frame lengths for each VL
  - *BAG* and S_{max} define the maximum bandwidth allocated to a VL
    - ★ Example: $BAG = 16$ ms and $S_{max} = 128$ bytes ⇒ 64000 bits/s
  - Scheduling of VLs on end systems ⇒ (bounded) jitter

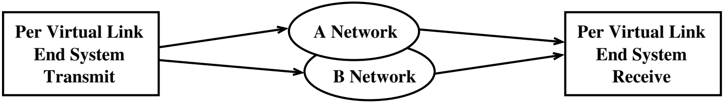


AFDX VL integration



The redundancy management in the AFDX

- Frames transmitted on two independent networks



- Identification of replicas via a sequence number

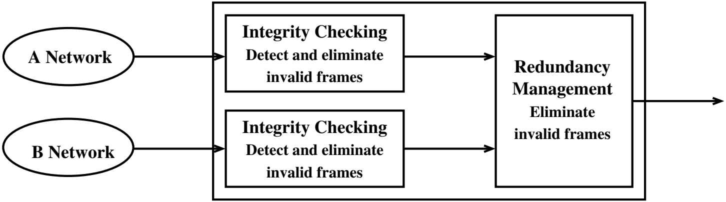
Ethernet Header	IP Header	UDP Header	Avionics Subsystem messages	Sequence number	FCS
14 bytes	20 bytes	8 bytes	17..1472 bytes	1 byte	4 bytes

UDP Header and payload

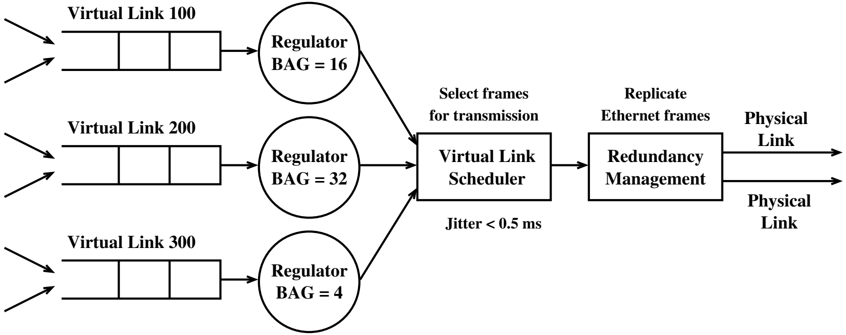
IP Header and payload

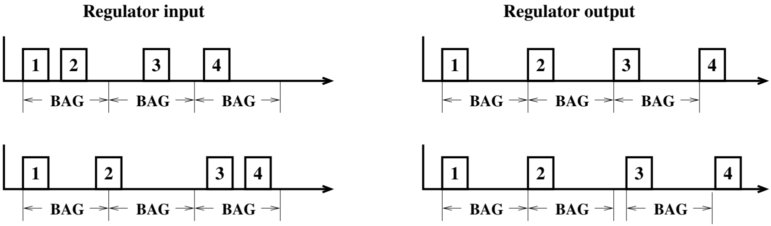
Ethernet frame

- Elimination of the second received frame



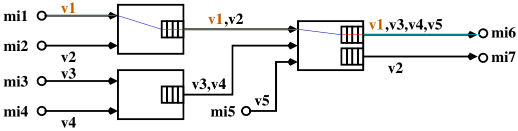
The packet regulation of Virtual Links





End-to-end delay on an AFDX network

- Three parts in the delay
  - Transmission times on links (known): link bandwidth, frame size
  - Switching latency (known)
  - Waiting time in FIFO queues (unknown)

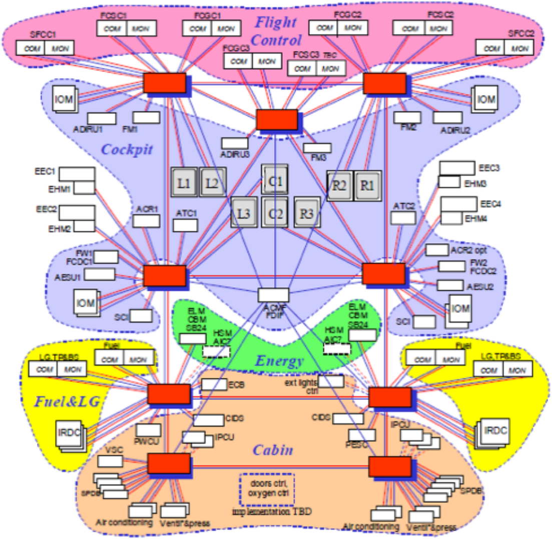


- Best-case delay: no waiting time in FIFO queues
- Worst-case delay: maximum overall waiting time in FIFO queues
- Distribution of the delay: distribution of the waiting time in FIFO queues

AFDX worst-case e2e delay analysis

- Mandatory for certification
- Different methods have been considered
  - Network calculus, used for the certification of Airbus aircraft, thanks to Confgen tool
    - Based on (min,+) algebra
    - Traffic is over-approximated
    - Service is under-approximated
    - Sure (pessimistic) upper-bound of end-to-end delays
    - Sure (pessimistic) upper-bound on buffer occupation
  - Trajectory approach
    - Worst-case interference for a frame on its trajectory
    - Some pessimistic over estimation
  - Model checking
    - Exhibit the worst-case scenario
    - Combinatorial explosion problem

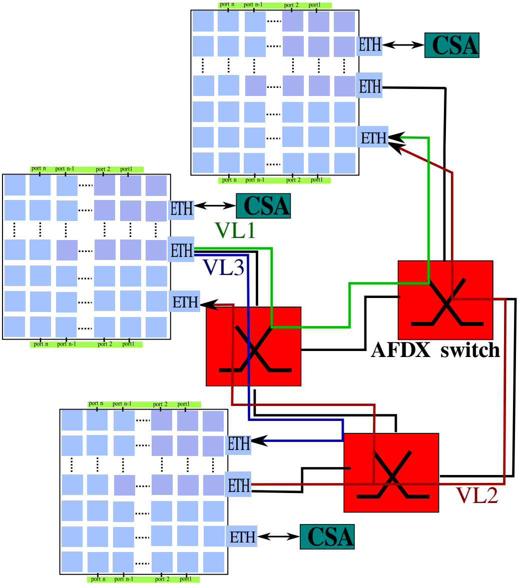
A typical avionics architecture



Summary

1. From classical avionics to IMA
2. Distributed manycore architectures
3. Avionics and non avionics flows on the same network
4. Conclusion

Envisioned avionics architecture



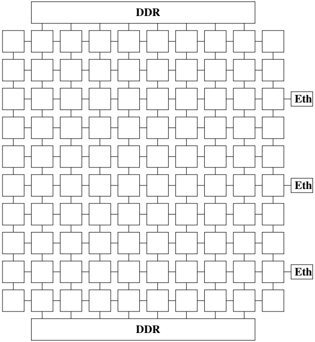
Envisioned avionics architecture

- Each manycore replaces several avionics computers
- A manycore: simple cores interconnected by a NoC
- Mapping of avionics applications on these manycores
- Data exchanges
  - between functions within a manycore
  - between functions on different manycores

Problem

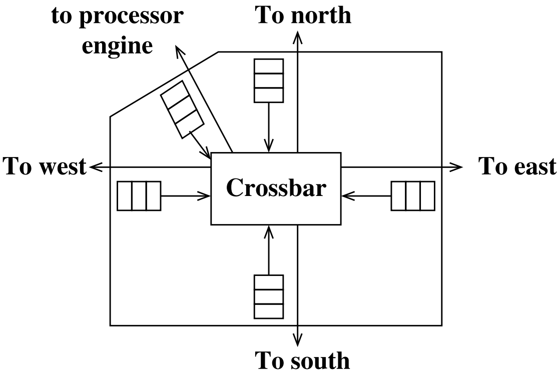
Find a mapping which minimizes communication latencies and jitters within manycores

The Tilera Tile64 manycore



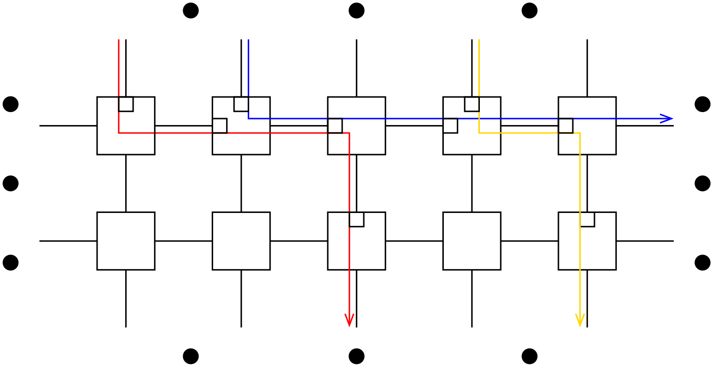
- X-Y wormhole switching with flow control in each router
- very small buffers in routers

The Tilera Tile64 router



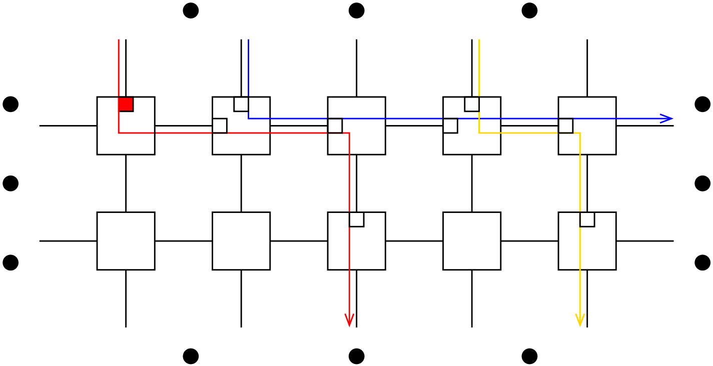
- Round robin arbitration between input ports

Wormhole switching



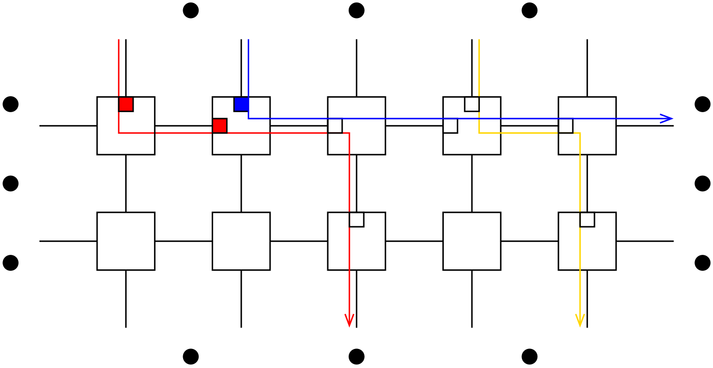
- Three flows, three flit packets

Wormhole switching



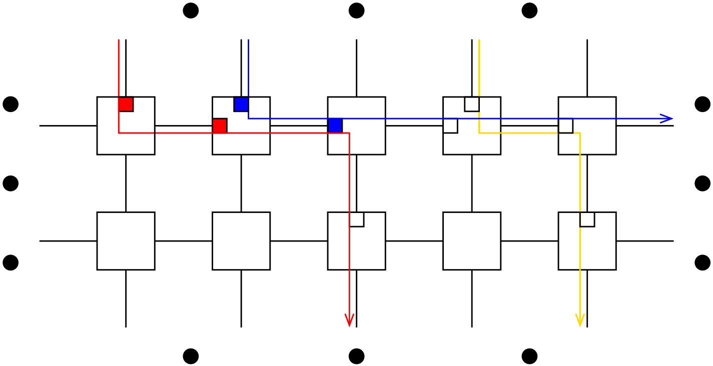
- Red flow: first flit in first router on its path

Wormhole switching



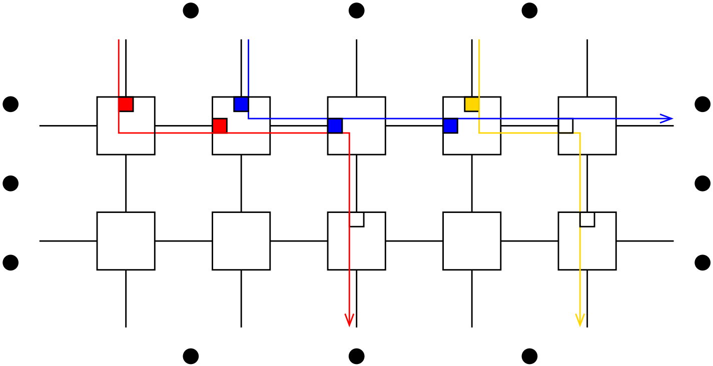
- Red flow: two first flits in two first routers on its path
- Blue flow: First flit in first router on its path

Wormhole switching



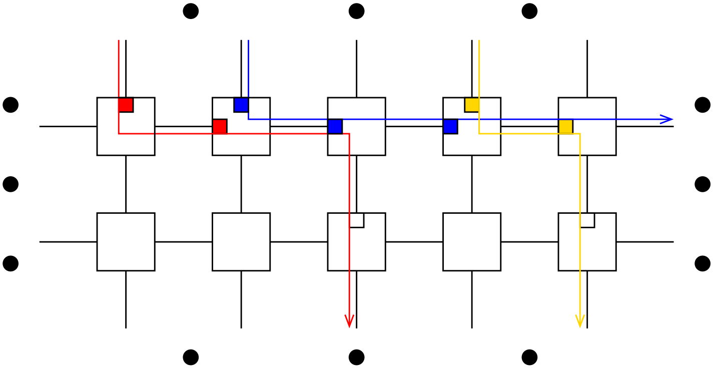
- Red flow directly blocked by blue flow

Wormhole switching



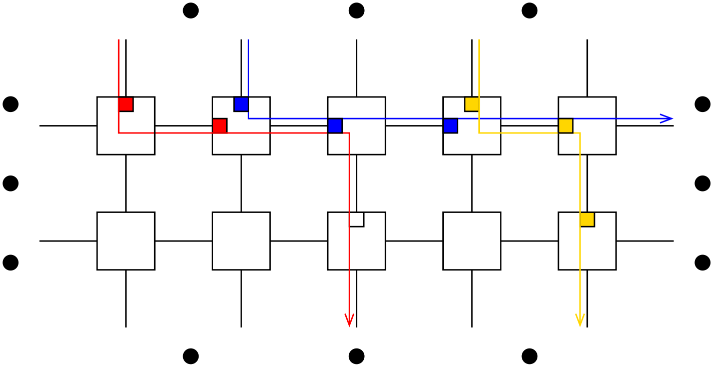
- blue flow progresses to next router on its path

Wormhole switching



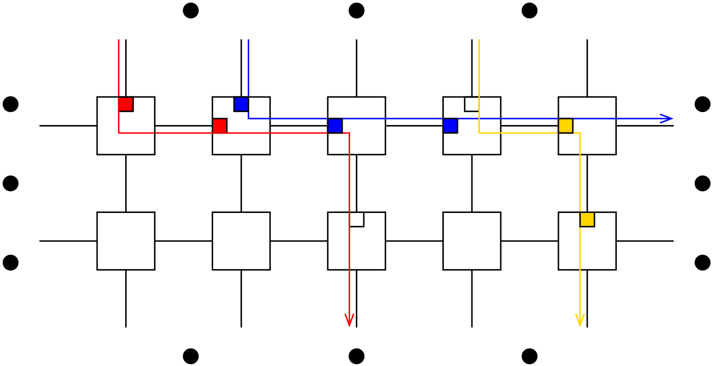
- blue flow directly blocked by yellow flow ⇒ red flow indirectly blocked by yellow flow

Wormhole switching



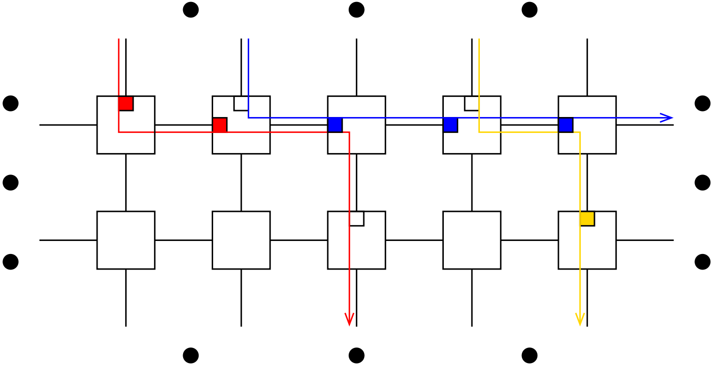
- One step for yellow flow

Wormhole switching



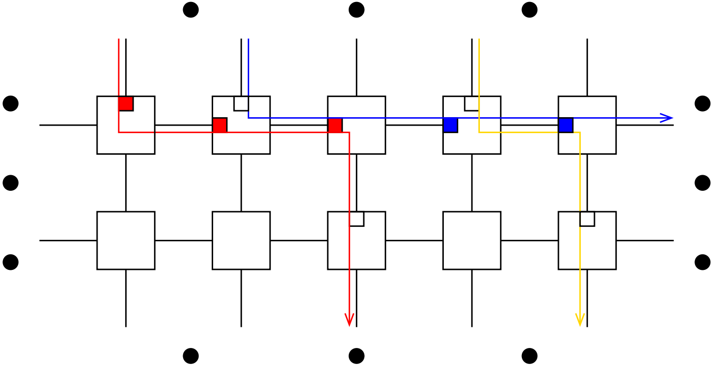
- One step for yellow flow

Wormhole switching



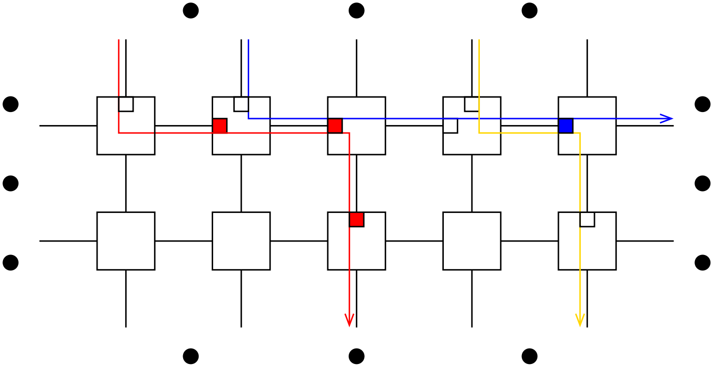
- One step for yellow flow
- One step for blue flow (no more blocked)

Wormhole switching



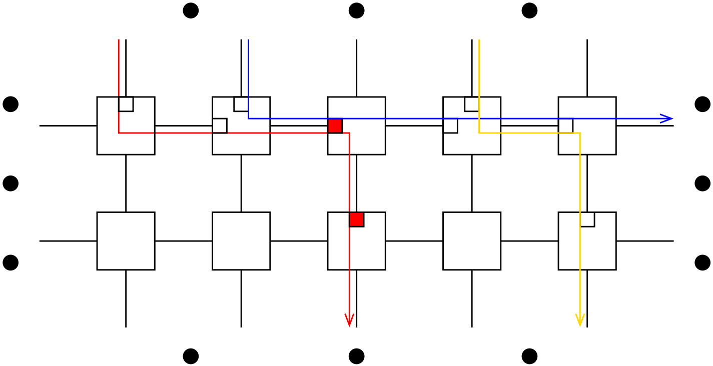
- One step for blue flow
- One step for red flow (no more blocked)

Wormhole switching



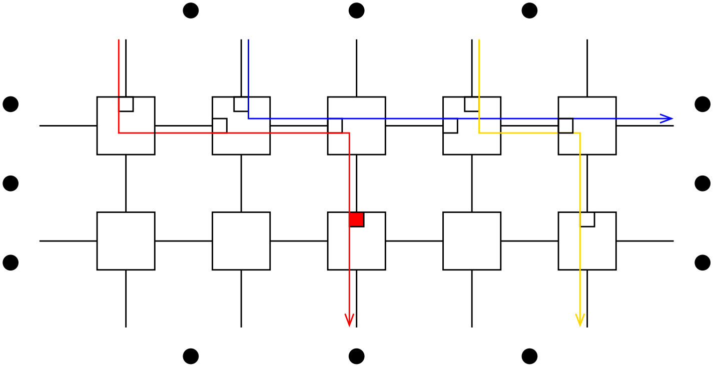
- One step for red flow

Wormhole switching



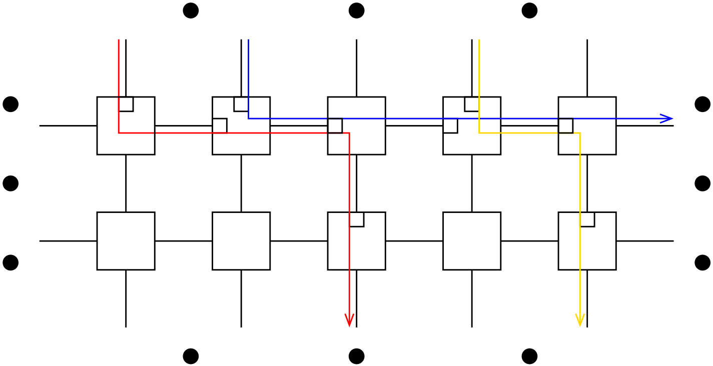
- One step for red flow

Wormhole switching



- One step for red flow

Wormhole switching



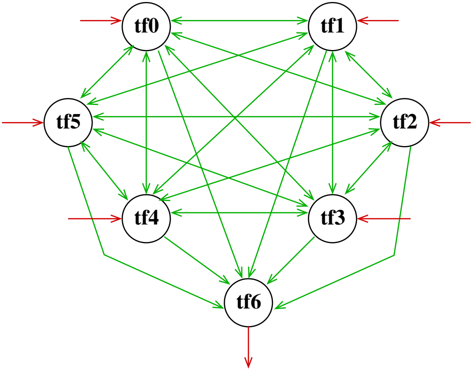
- One step for red flow

Worst-case delay analysis for Tilera-like NoCs

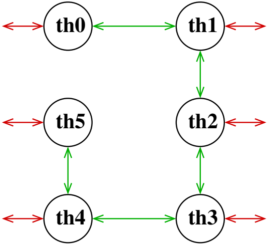
- Two sets of impact
  - Direct blocking
    - Red flow can be directly blocked by blue one
    - Blue flow can be directly blocked by yellow one
  - Indirect blocking
    - Red flow can be indirectly blocked by yellow one
- A state-of-the-art worst-case analysis: Recursive Calculus
  - Identify all the flows which share an output port with the flow under study
  - For each identified flow, compute its worst-case delay to reach its destination

Typical avionics applications

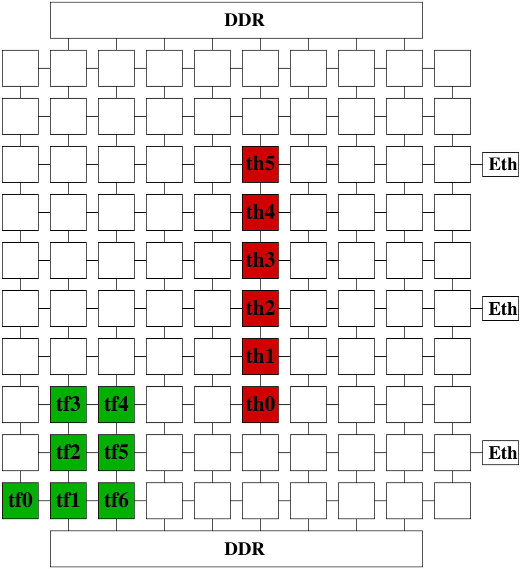
Full Authority Digital Engine (FADEC)
Critical application



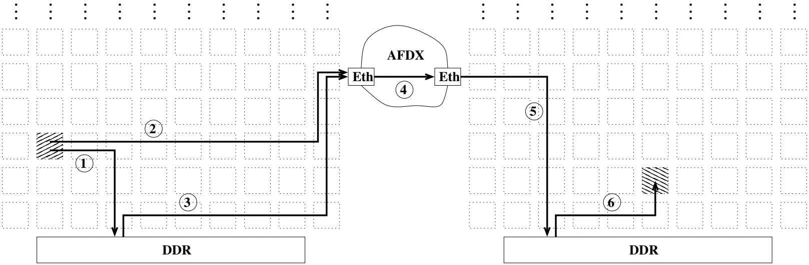
Health Monitoring (HM)
Non critical application



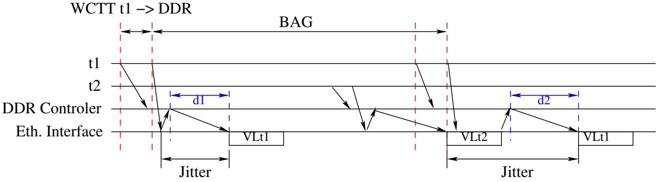
Mapping of avionics applications on manycore



Inter manycore communications



VL transmission

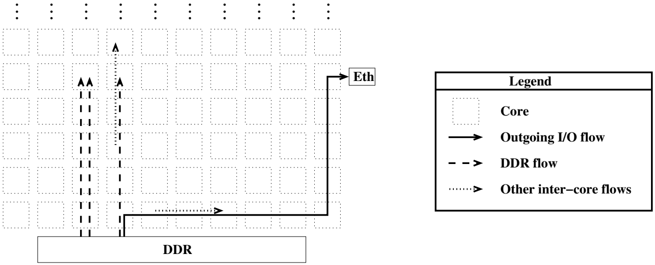


- Jitter for the DDR → Ethernet part
- Guaranteed upper bound of 500 μs for this jitter (certification constraints)

State-of-the-art mapping strategies

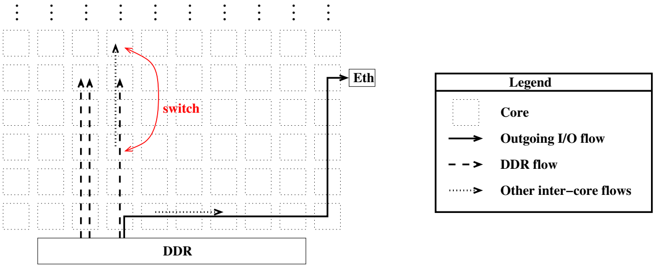
- SHiC: reduces contentions on the core-to-core communications
  - Application allocation: Search for regions which size = size of the application ⇒ no fragmented regions
  - Tasks allocation: minimum distance between communicating tasks
  - Inter manycore communications not taken into account
- MAP$_{io}$: Take into account input flows
  - Allocates primarily critical applications in a region close to memory and Ethernet controllers
  - Minimize contentions on the paths of input flows
- No strategy takes into account output flows

Additional rules to minimize contentions for an output flow



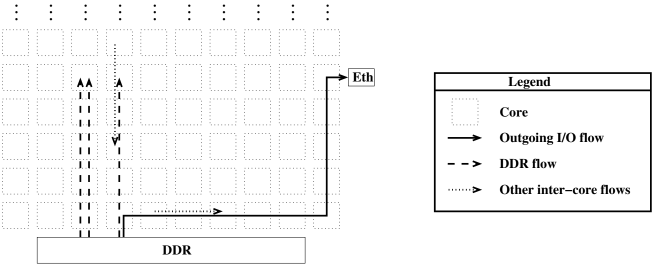
- Minimize DDR flow impact ⇒
  - source task of output flows in columns with minimum DDR usage
  - Map tasks in order to minimize inter-core flows in the same direction as DDR ones

Additional rules to minimize contentions for an output flow



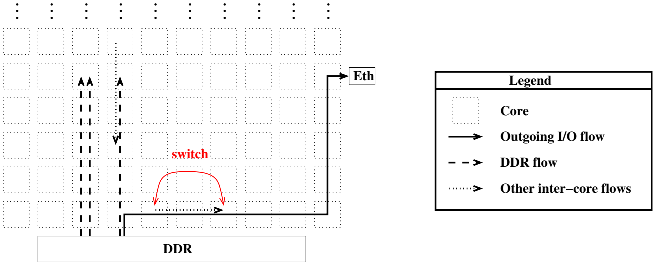
- Minimize DDR flow impact ⇒
  - source task of output flows in columns with minimum DDR usage
  - Map tasks in order to minimize inter-core flows in the same direction as DDR ones

Additional rules to minimize contentions for an output flow



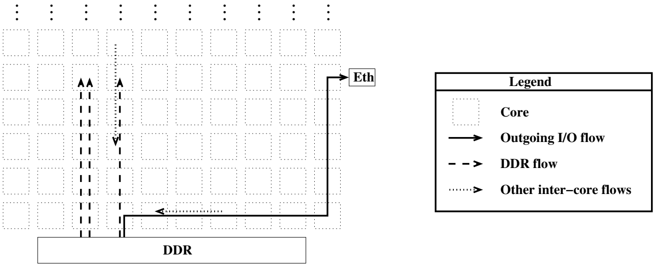
- Minimize inter-core flow impact ⇒
  - Map tasks in order to minimize inter-core flows in the same direction as output ones

Additional rules to minimize contentions for an output flow



- Minimize inter-core flow impact ⇒
  - Map tasks in order to minimize inter-core flows in the same direction as output ones

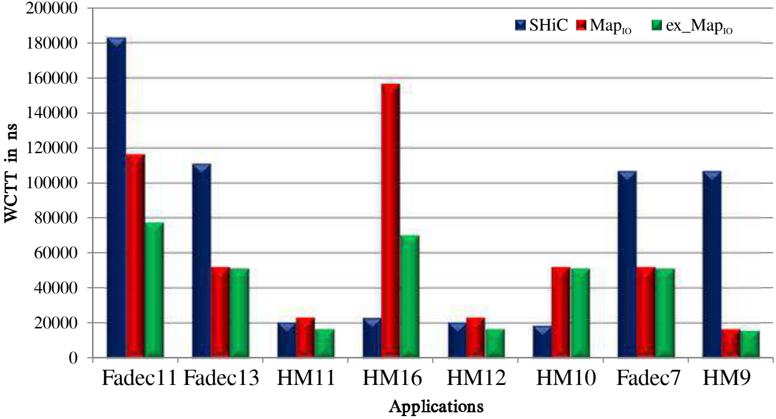
Additional rules to minimize contentions for an output flow



- Minimize inter-core flow impact ⇒
  - Map tasks in order to minimize inter-core flows in the same direction as output ones

Results

- Worst-case analysis using state-of-the-art Real-Time Calculus



Summary

1. From classical avionics to IMA

2. Distributed manycore architectures

3. Avionics and non avionics flows on the same network

4. Conclusion

Motivation

- Certification constraints
  - No missed deadline for any flow
  - No frame loss due to buffer overflow
- Suppliers are allocated VLs which over-provision their needs

⇒ The AFDX network is lightly loaded

- Other technolgies are used to transmit non avionics flows
  - Audio for the crew
  - Video (cameras for parking guidance, monitoring system)
  - ...
- Increase weight, cabling, ...
- Use spare AFDX bandwidth to transmit part or all of these non avionics flows
- Guarantee flow constraints
- Constraints depend on flow classes ⇒ Differentiated treatment of flows ⇒ Quality of Service facilities

Quality of Service facilities

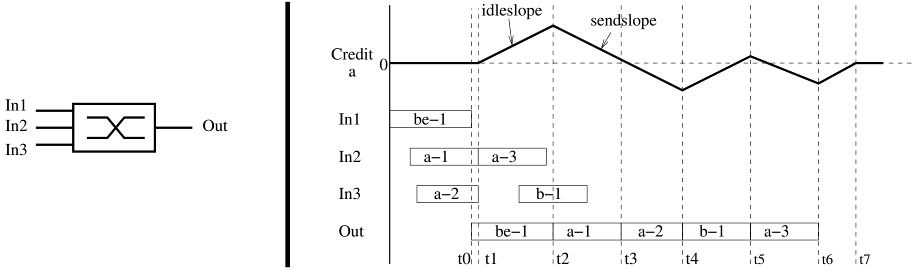
- Many existing service discipline (not exhaustive)
  - First In First Out
    - Frames treated in their order of arrival
  - Static Priority Queueing
    - Each flow is assigned a priority
    - Frames scheduled in each output port, strictly following these priorities
  - Round Robin
    - Each flow is assigned a class
    - Classes are polled in round robin order in each output port
    - Each class is allocated a credit: number of frames (e.g. Weighted Round Robin) or number of bytes (e.g. Deficit Round Robin)
  - Fair Queueing
    - Each flow is allocated a class
    - Each class is allocated a percentage of the bandwidth
    - Scheduling should be as close as possible as a perfect sharing
    - WFQ, WF^{2}Q, . . .

Real time Ethernet technologies

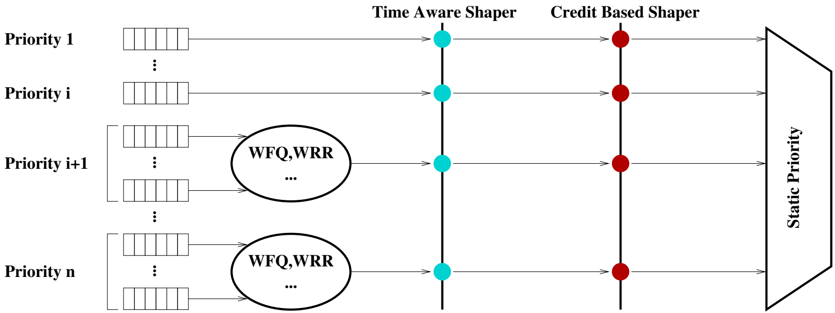
- TTEthernet
  - Three types of traffic
    - Time-Triggered: reserved slots
    - Rate Constrainted: AFDX-like
    - Best effort
  - Global synchronization
- Ethernet-AVB
  - Priority queueing
  - Credit Based Shapers for 2 highest priorities, to avoid starvation problems
- Time Sensitive Networking
  - Extension of AVB for control data traffic
  - A higher priority class with reserved slots
  - Based on Time Aware Shapers

Illustration of Credit Based Shaper

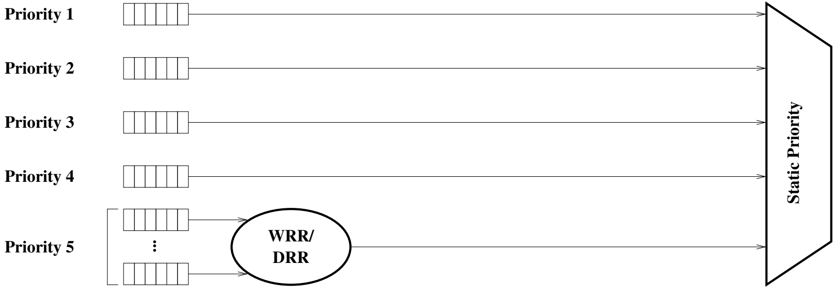
- Three classes of traffic: a (high priority), b (average priority), be (low priority)
- A Credit Based Shaper associated with class a



Synthesis of potential QoS facilities in a switch output port

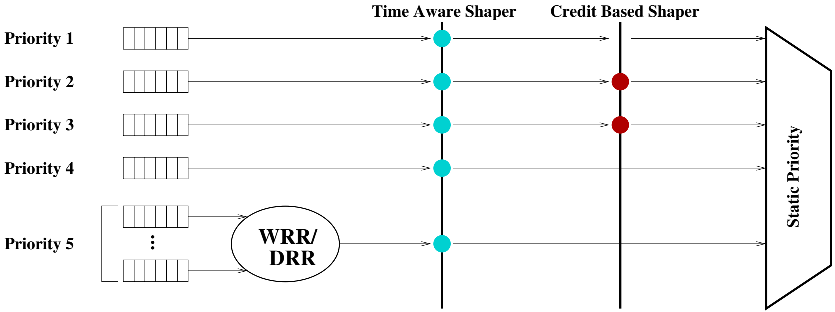


Candidate solution 1



- 4 highest priorities for avionic flows
- Priority 5 shared by non avionic flows

Candidate solution 2



- Highest priority for critical control flows
- Priority 2, 3, 4 for other avionics flows
- Priority 5 shared by non avionic flows

Temporal analysis

- Hard constraints for avionic flows
- Soft constraints for non avionic flows
  - e.g. Quality of Experience for video
  - Average behavior, probabilistic guarantees
- Worst-case analysis is mandatory for avionic flows
  - Existing analysis for different QoS facilities, more or less pessimistic
- Worst-case analysis is most of the time not the right solution for non avionics flows
- Simulation is often sufficient

Summary

1. From classical avionics to IMA
2. Distributed manycore architectures
3. Avionics and non avionics flows on the same network
4. Conclusion

Conclusion

- Two main evolution of avionics architecture considered
  - Integration of manycores in an AFDX architecture
  - Transmission of additional non avionic flows on the AFDX
- Enhanced mapping strategy in order to take into account inter-manycore flows and jitter constraints
- Candidate QoS solutions

Some next steps

- End-to-end analysis (NoC + AFDX)
- Consider other options such as a core dedicated to input/output flow management
- Mapping of flows on QoS facilities
- Tuning of QoS facilities

Thank you for your attention!