KALRAY

Kalray MPPA®

Massively **P**arallel **P**rocessor **A**rray

Service Guarantees for the Wormhole Switching NoC of the MPPA-256 Bostan

Benoît Dupont de Dinechin, CTO

Outline

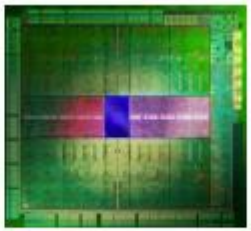
- MPPA®-256 Bostan Processor
- MPPA® Application Environment
- MPPA® Bostan NoC Architecture
- Max-Min Feed-Forward Routing
- Deterministic Network Calculus (DNC)
- DNC Application to the MPPA® Bostan NoC
- Conclusions

MPPA® Manycore Processors

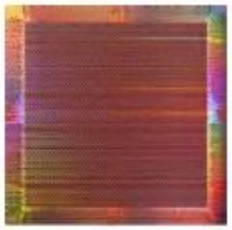
- DSP type of processing
  - Energy efficiency
  - Timing predictability
  - Software programmability
- CPU ease of programming
  - C/C++ GNU environment
  - 32-bit/64-bit addresses, little-endian
  - Rich operating systems (Linux with dynamic loading & linking)
- MPPA®-256 processors
  - 32 management cores on chip
  - 256 application cores on chip
  - High-performance I/O
- Scalable parallel computing
  - MPPA® processors can be tiled together through NoC
  - Run-time support for pooling the external DDR memory resources

CPUs

DSPs



GPUs



FPGAs

Altera

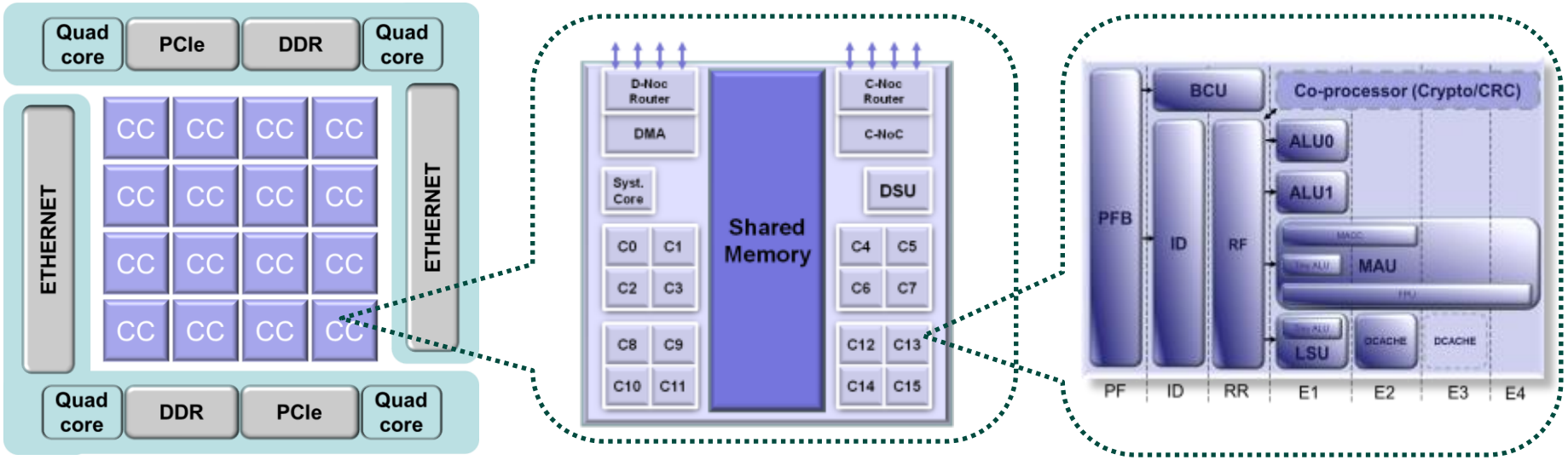
Courtes

MPPA® Processor Family and Roadmap

Production 2015 2017 2018

	MPPA® 1.0 ANDEY	MPPA® 2.0 BOSTAN 288	MPPA® 2.5 BOSTAN-S 288	MPPA® 3.0 COOLIDGE 80 /COOLIDGE160	
	Prototype	Higher speed & performance Adding new features : Ethernet dispatcher, Crypto co-processor		Optimized for Computer Vision and Deep Learning	
Fixed-point operations	**600 GOPS**	**1 TOPS**		**6 TOPS**	**9..2 TOPS**
Floating-point operations	**600 GFLOPS**	**1 TFLOPS**		**3.0 TFLOPS**	**4.6 TFLOPS**
Typical Power	**10 – 15 W**	**10 – 20 W**		**6 – 20 W**	**10W – 40W**
Specifications	• 288 VLIW Cores • 32 bits • 2xDDR3 • 2X40 GbE • 2xPCIe 8 lane Gen3	• 288 Kalray VLIW Cores • 128 Crypto Coporcessors • 2xDDR3 • 8x 1/10G GbE • 2xPCIe 8 lane Gen3		• 80 Kalray 64-bit VLIW Cores • 80 Coprocessor for vision and learning • 2 x LP/DDR4 • 8x 1/10/25GbE • 16-lane PCIe Gen4 • 4x CAN-FD Controller	• 160 Kalray 64-bit VLIW Cores • 160 Coprocessor for vision and learning • 2 x LP/DDR4 • 8x 1/10/25GbE • 16-lane PCIe Gen4 • 4x CAN-FD Controller

MPPA®-256 Bostan Processor Architecture



Manycore Processor

- 16 compute clusters
- 2 I/O clusters each with quad-core CPUs, DDR3, 4 Ethernet 10G and 8 PCIe Gen3
- Data and control networks-on-chip
- Distributed memory architecture
- 634 GFLOPS SP for 25W @ 600Mhz

Compute Cluster

- 16 user cores + 1 system core
- NoC Tx and Rx interfaces
- Debug & Support Unit (DSU)
- 2 MB multi-banked shared memory
- 77GB/s Shared Memory BW
- 16 cores SMP System

VLIW Core

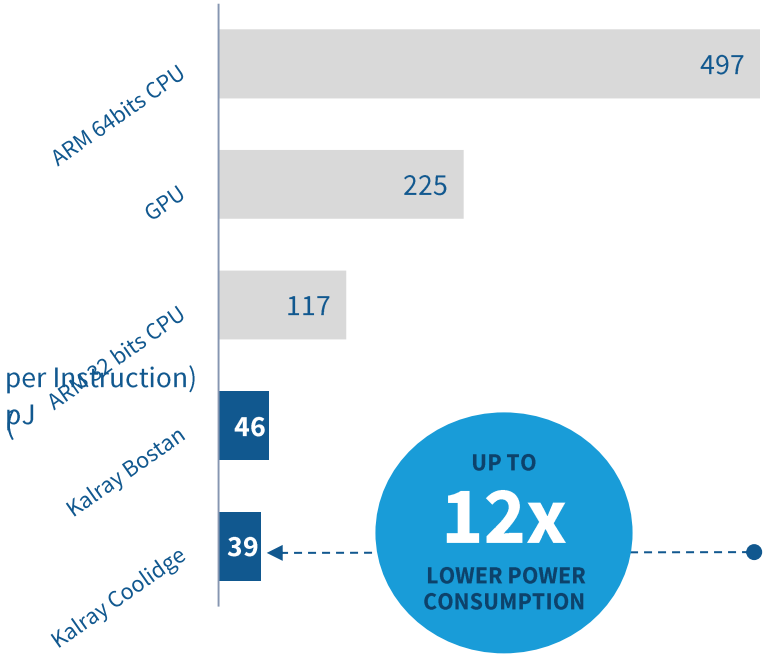
- 32-bit or 64-bit addresses
- 5-issue VLIW architecture
- MMU + I&D cache (8KB+8KB)
- 32-bit/64-bit IEEE 754-2008 FMA FPU
- Tightly coupled crypto co-processor
- 2.4 GFLOPS SP per core @600Mhz

From Multicore to MPPA® Manycore Processor

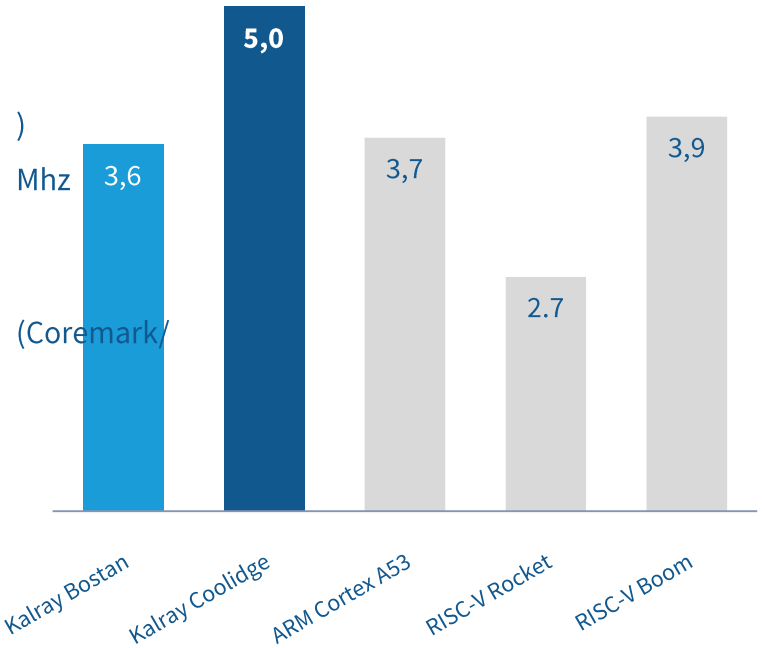
- Manycore consolidates embedded multicores
  - Multicore units with local memory connected by a NoC
  - Tightly coupled memory (TCM) accesses are more energy-efficient than a Last-Level Cache (LLC)
  - Local memory access interferences can be managed for time-critical computing
- Asynchronous one-sided communication (DMA)
  - Generalization of OpenCL `async_work_group_copy`
  - Zero-copy, non-blocking data transfers like in MPI-3
  - Accesses to other cluster local memories and external DDR memories through remote DMA over NoC
  - Coordination through remote atomic operations and remote queues
- Load/store accesses to external memory
  - Mostly used for CPU application software porting
  - Less efficient than DMA, not time-predictable
  - Required for load/store external memory accesses that cannot be converted to asynchronous data transfers

KALRAY VLIW CORES
MPPA BOSTAN AND MPPA COOLIDGE

Energy efficiency benchmark

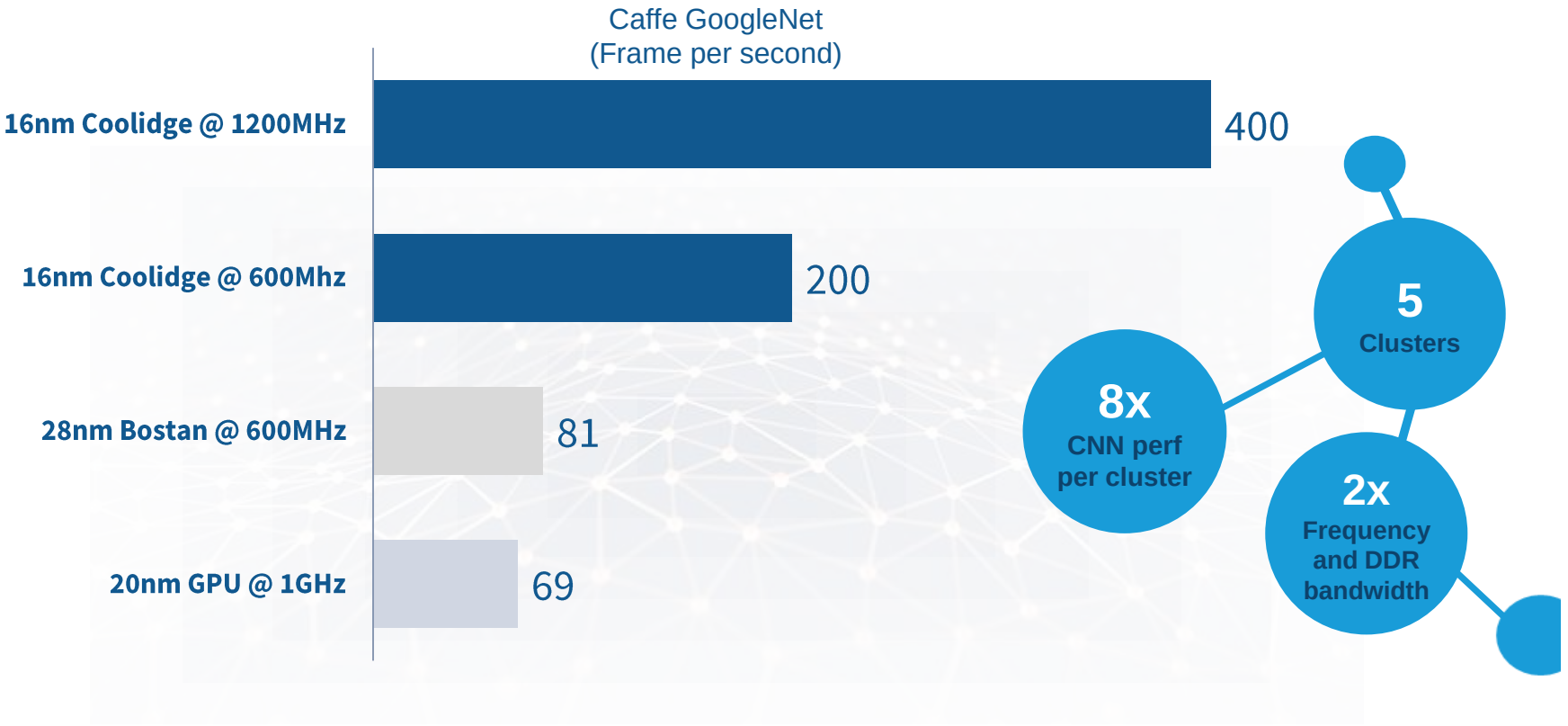


Coremark/MHz benchmark



(1) Source: "An Instruction Level Energy Characterization of ARM Processors", ARM Cortex, FORTH-ICS/TR-450, March 2015 (ARM Cortex A15 & ARM Cortex A7)
(2) Source: Measured on Matrix Multiply EPI Test + Source Bill Dally, "To ExaScale and Beyond" – Nvidia

MPPA® BOSTAN VS COOLIDGE ON CNN INFERENCE



(*) Half Precision FLOPS - 16 FMA/cycle/core with CNN co-processor
Including PCIe gen3 x8 – DDR4 3200 – Ethernet 4x1Gb

MPPA® Processor Co-Design for Avionics

- U. Saarland / AbsInt GMBH recommendations on VLIW core and cache micro-architecture design
  - AbsInt provides the aiT static timing analysis tool used to certify flight control systems at Airbus
  - AbsInt aiT tool targets the Kalray VLIW cores
- Processor design with a focus on timing predictability
  - Core level: micro-architecture
    - ✓ Fully timing compositional core
    - ✓ LRU caches and write buffer
    - ✓ Cache bypass memory loads
  - Cluster level: multi-banked shared memory
    - ✓ Core-private buses for memory bank access
    - ✓ Address interleaving or blocking across banks
  - Processor level: NoC/DDR guaranteed services
    - ✓ NoC minimum bandwidth & maximum latency
    - ✓ DDR controller configurations and address mapping

Certification of Real Time Applications desIgNed for mixed criticaliTY

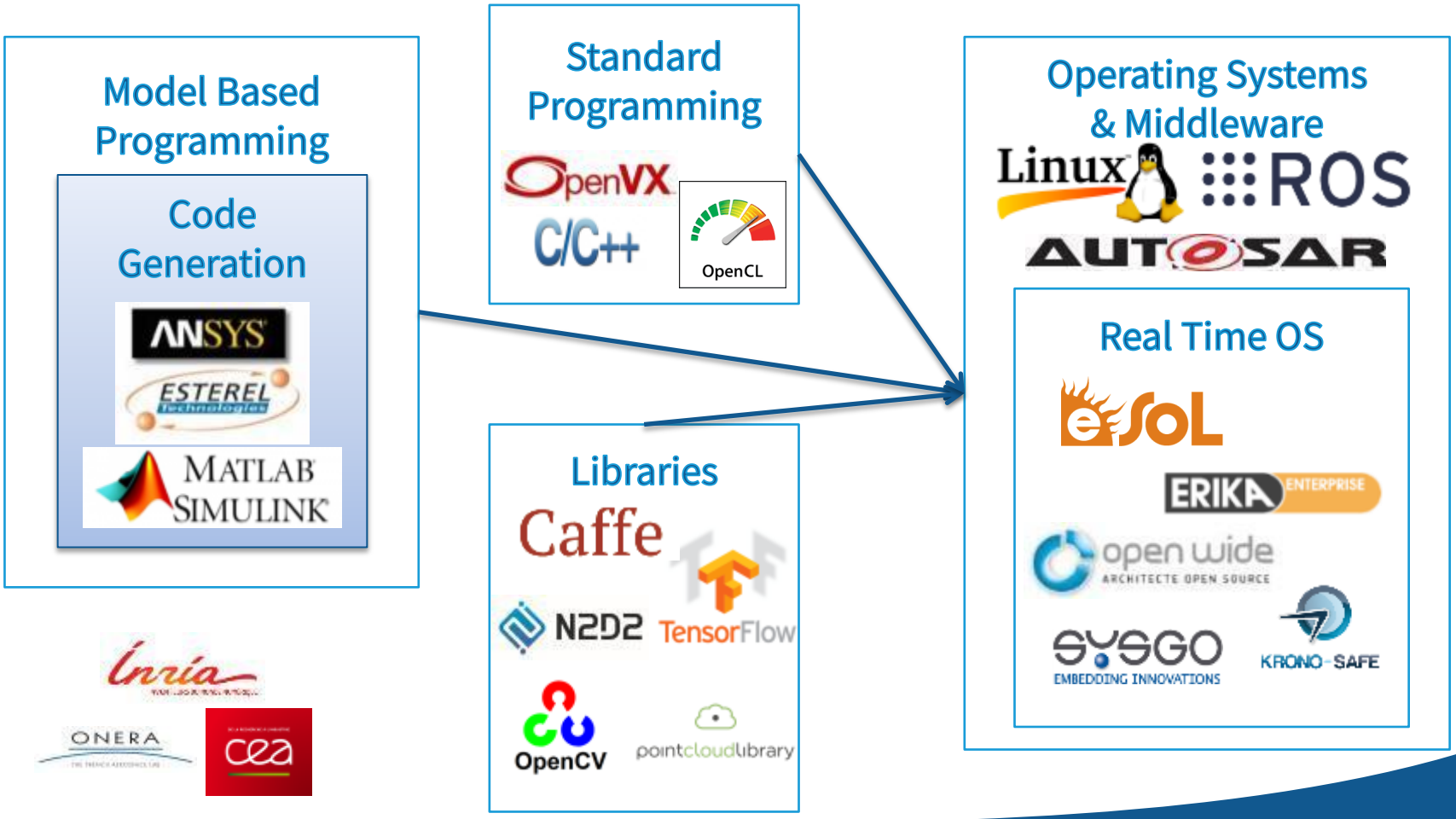
Outline

- MPPA®-256 Bostan Processor
- MPPA® Application Environment
- MPPA® Bostan NoC Architecture
- Max-Min Feed-Forward Routing
- Deterministic Network Calculus (DNC)
- DNC Application to the MPPA® Bostan NoC
- Conclusions

MPPA Processing Platform

- A processing platform is the combination of the processor architecture and basic software exposed to applications
- MPPA applications are contained into execution domains
  - Set of execution resources isolated from each other by locked out configurations
  - Applications in different domains can only interact through external interfaces (PCIe, Ethernet)
- Each MPPA domain may be assigned a criticality level
  - Hard real-time (typically time-triggered) (SCADE Suite, Simulink)
  - Soft real-time (typically event-triggered) (Dataflow, OpenVX)
  - Best effort (typically high-performance) (OpenCL, OpenMP)
- Inside a domain, the Kalray mOS hypervisor based on exo-kernel principles enforces robust spatial partitioning
  - Domains are decomposed into one or more partitions
  - Partitions communication mechanisms are mediated by mOS

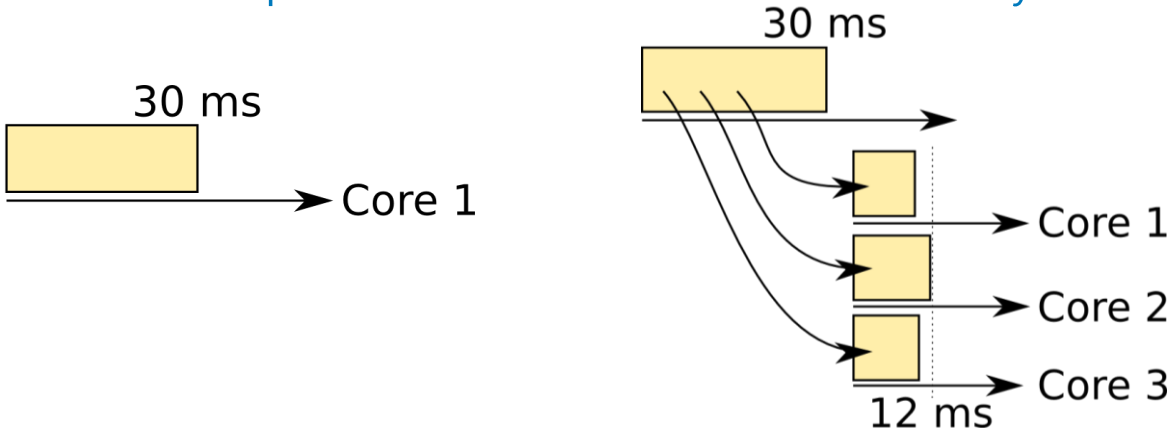
MPPA SOFTWARE ECOSYSTEM FOR SAFE & EFFICIENT APPLICATIONS



KALRAY

SCADE Code Generation for the MPPA®

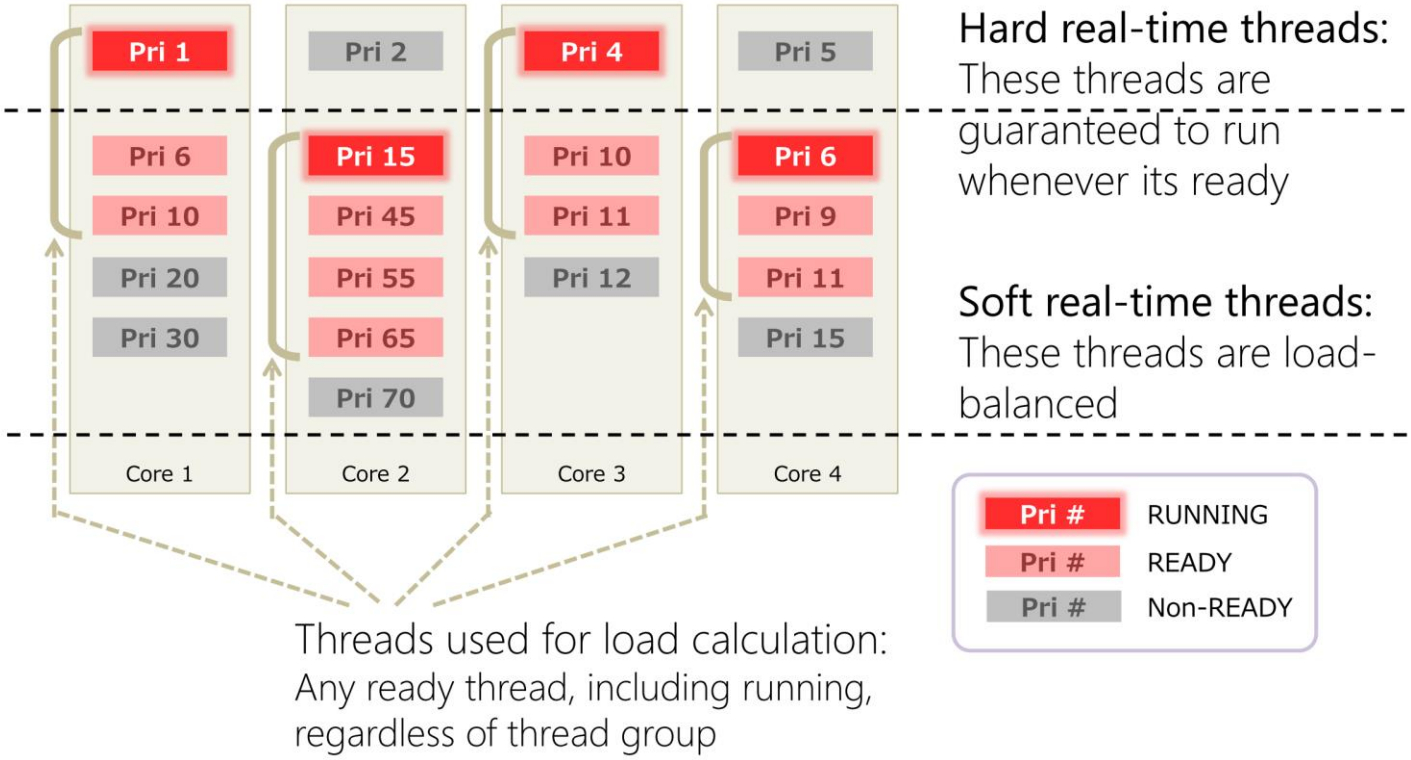
- Safety-critical control-command applications
  - Model-based programming using SCADE Suite® from Esterel Technologies
  - Complemented with static timing analysis of binary code (aiT from AbsInt)
  - Retargeting of the formally proven bug-free CompCert C99 compiler
- Motivations for multicore and manycore execution
  - Distribute the compute load across cores and reduce memory interferences



  - Effective implementation of multi-rate harmonic applications
  - Envision use of fast Model Predictive Control (MPC) techniques

eMCOS for MPPA® Processors

- eSOL eMCOS is the world's first many-core RTOS for embedded use
  - Runs on the MPPA® clusters on top of mOS exo-kernel



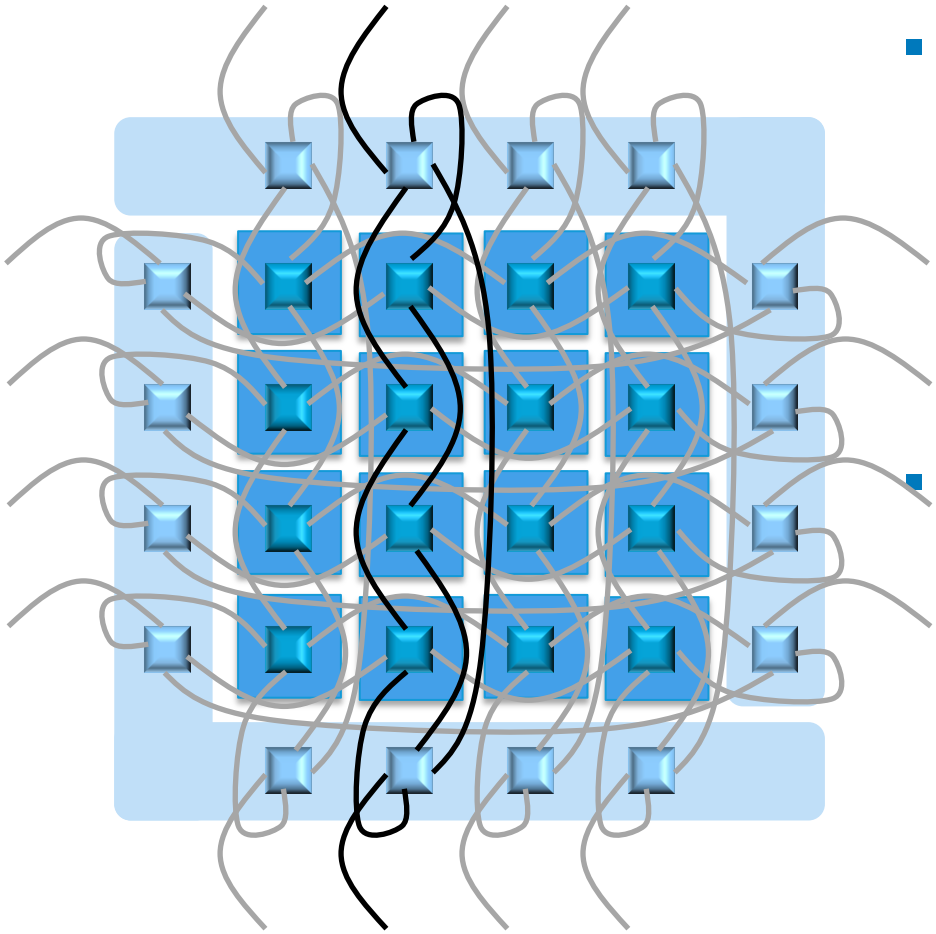
MPPA® Extensions of OpenCL 1.2

- Parts of standard OpenCL that are useful on a manycore processor
  - Host program allocates global buffers, creates executables kernels, and dispatches work in queues
  - Kernel invocation with a user-defined argument list, which distinguishes between local and global objects
- OpenCL extensions for CPU-based manycore platforms
  - Kernel code written in standard C/C++ and/or assembly language
  - Kernel code with CPU multi-threading [TI's "OpenMP Dispatch With OpenCL"]
  - Kernel code that accesses the local memory of other Compute Units
- OpenCL 1.2 extensions for the MPPA® processor
  - Use the OpenCL Task Parallel mode to dispatch one Work Group of one Work Item on each cluster
  - Kernel code linked with ELF executable uses Pthreads or GCC OpenMP to activate cluster cores
  - Kernel code accesses the full asynchronous one-sided communications & synchronizations API

Outline

- MPPA®-256 Bostan Processor
- MPPA® Application Environment
- MPPA® Bostan NoC Architecture
- Max-Min Feed-Forward Routing
- Deterministic Network Calculus (DNC)
- DNC Application to the MPPA® Bostan NoC
- Conclusions

MPPA®-256 Bostan Network-on-Chip (NoC)



- Dual 2D-torus NoC
  - Direct network, wormhole switching
  - D-NoC: high-bandwidth RDMA
  - C-NoC: low-latency mailboxes
  - 4B/cycle per link direction per NoC
  - Nx10Gb/s NoC extensions for connection to FPGA or other MPPA®
- Rate-based QoS
  - Source routing must ensure deadlock-free traffic
  - Data NoC is configured by selecting routes and injection parameters
  - Injection parameters are the (σ,ρ) or (burst, rate) of network calculus

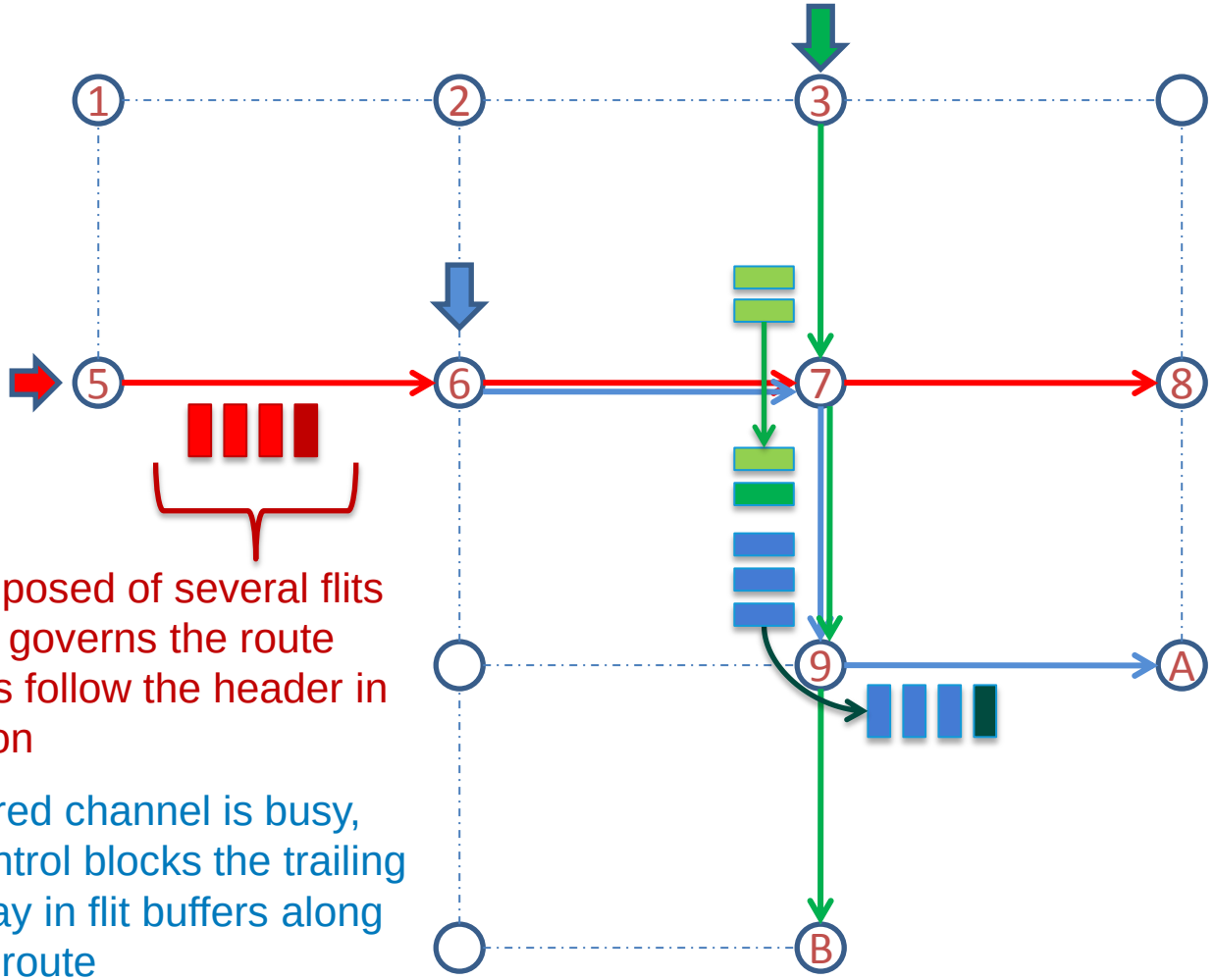
Interconnection Network Concepts

- Topology
  - How the nodes are connected together
  - Direct network if routing nodes can be endpoints
- Switching
  - Allocation of network resources (bandwidth, buffer capacity, …) to information flows
- Flow control
  - How a downstream node forwards availability to an upstream node
  - Applies at hop level, entry-to-exit level, and transport level
- Routing
  - Path selection between a source and a destination node in a particular topology

MPPA®-256 Bostan NoC Switching

- Network switching techniques
  - Circuit switching: network resources are dedicated over an end-to-end path before transmission starts
  - Packet switching:
    - Store and forward: node buffers entire packet before forwarding
    - Virtual cut-through: node starts forwarding as soon as buffer space for a whole packet is available on the next node
    - Wormhole switching: the packet is decomposed into flits that travel in a pipelined fashion, buffering is applied at flit level
- The MPPA® NoC implements wormhole switching with source routing and without virtual channels
  - A packet is composed of header flits and payload flits (32-bit flits)
  - The packet follows a route determined by a bit string in the header

Wormhole Switching Illustrated

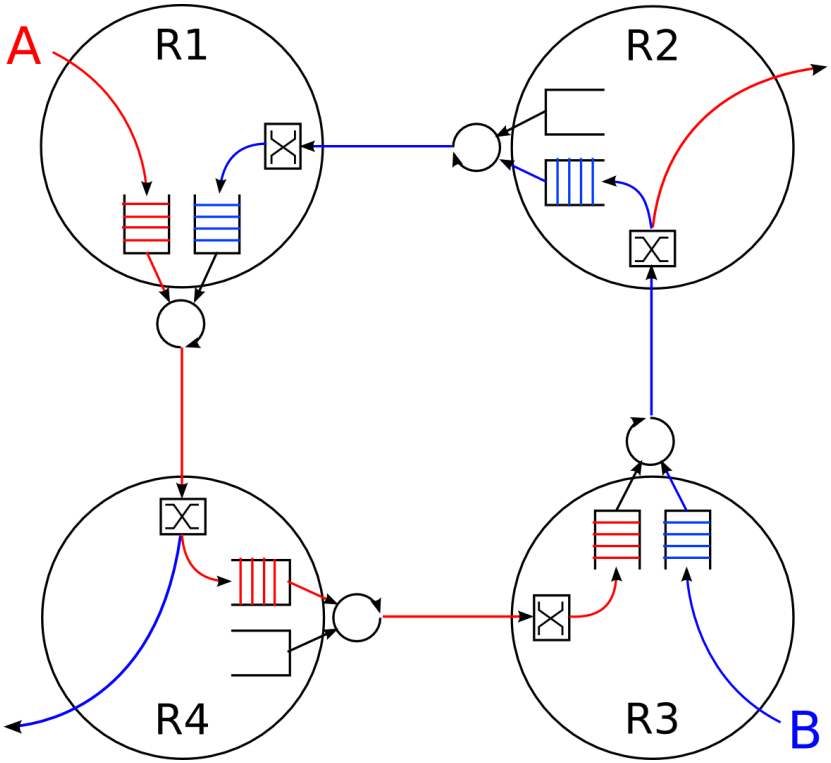


A packet is composed of several flits
The header flits governs the route
The payload flits follow the header in a pipeline fashion

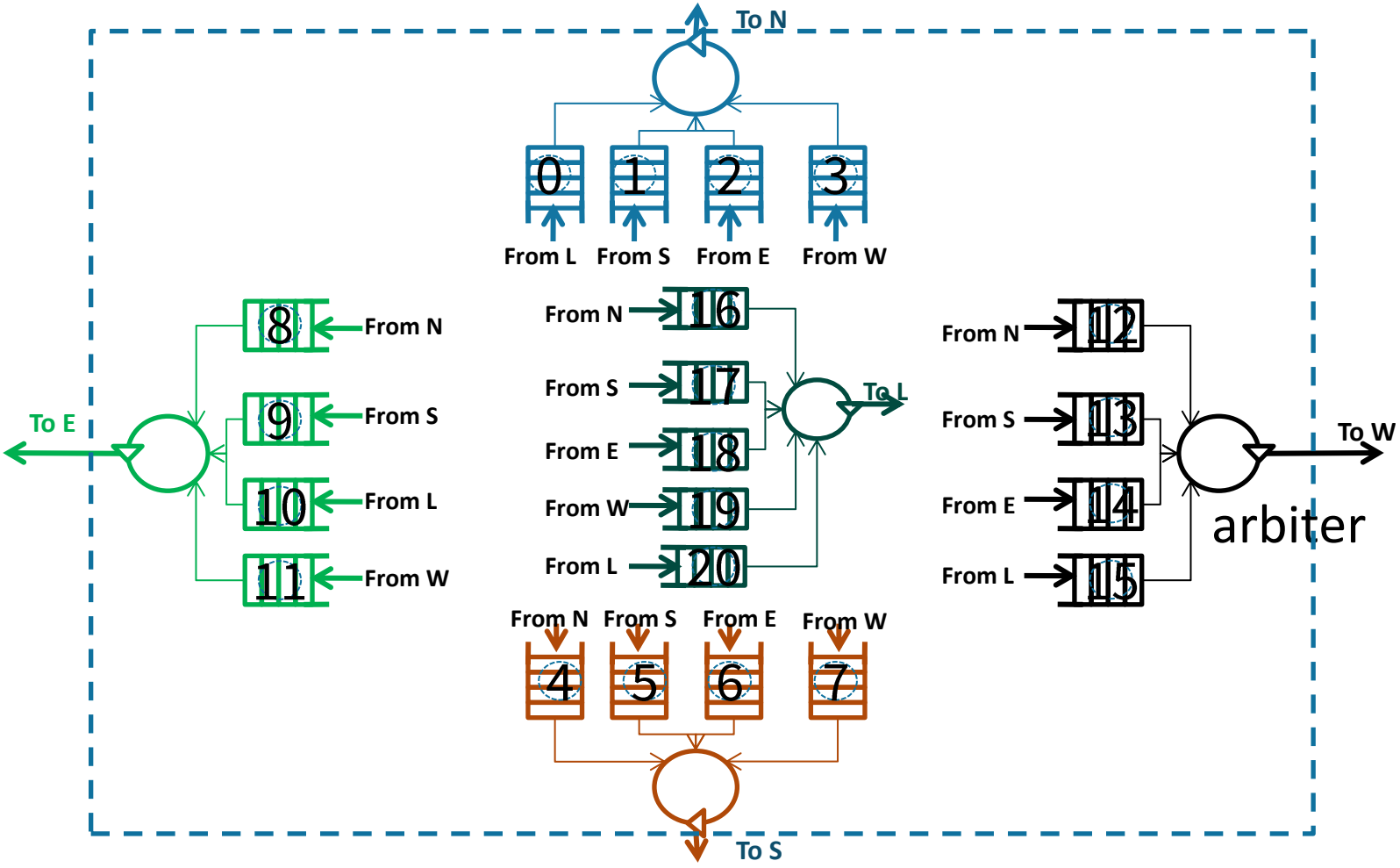
When the required channel is busy, the hop flow control blocks the trailing flits and they stay in flit buffers along the established route

Wormhole Switching NoC Issues

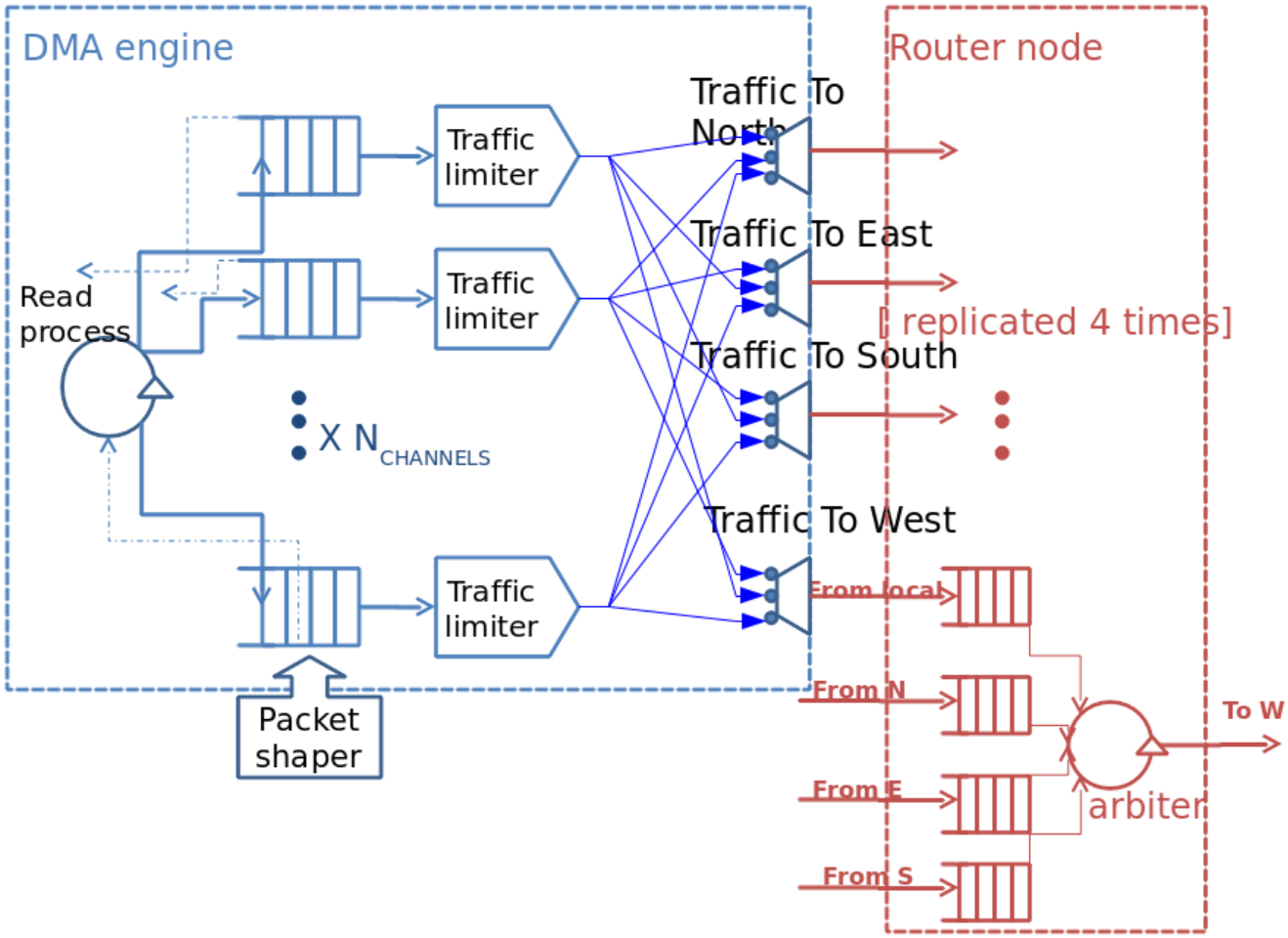
- Complex to implement
  - May be true for input queueing & output matching (e.g. iSLIP)
  - The MPPA® NoC routers only include demultiplexers, output queues and RR arbiters
- Prone to deadlocking
  - In this example, the red flow cannot use R3→R2 because the blue flow is using it
  - Likewise, the blue flow needs R1→R4 held by the red flow
  - Deadlock requires full queues



MPPA® NoC Router

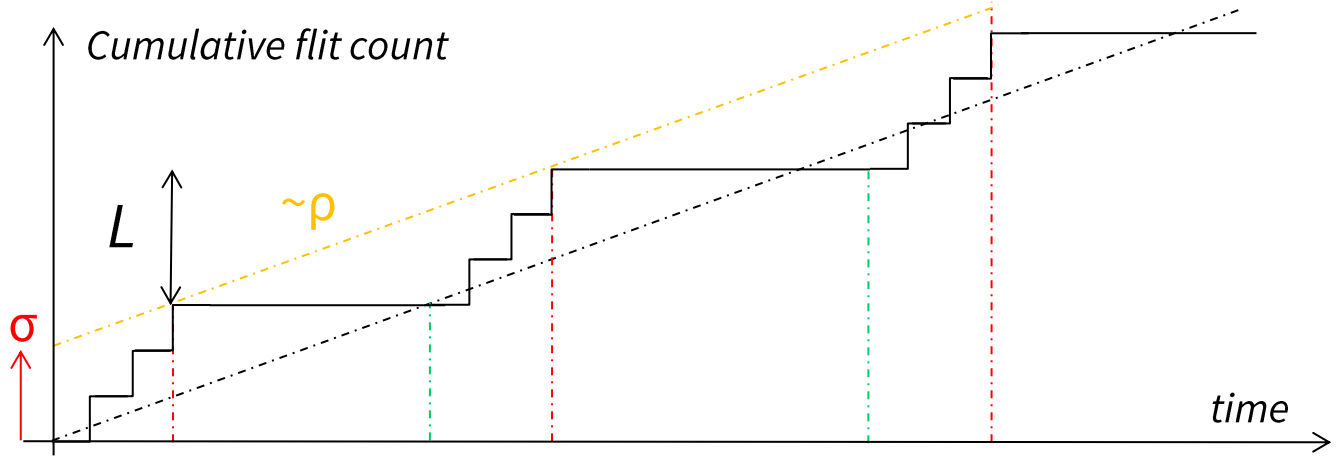


MPPA®-256 Data NoC Tx



Rate-Based MPPA® NoC Guaranted Services

- Data NoC packet injection implements a (σ,ρ) regulation
  - No more than σ+ρ(*t*-*s*) flits are injected for any interval [*s*,*t*]



- Application of Deterministic Network Calculus (DNC) prevents NoC congestion and provides bounds on end-to-end delays
- Application of DNC requires that flows be routed *feed-forward*

Outline

- MPPA®-256 Bostan Processor
- MPPA® Application Environment
- MPPA® Bostan NoC Architecture
- Max-Min Feed-Forward Routing
- Deterministic Network Calculus (DNC)
- DNC Application to the MPPA® Bostan NoC
- Conclusions

Constraints and Objectives of NoC Routing

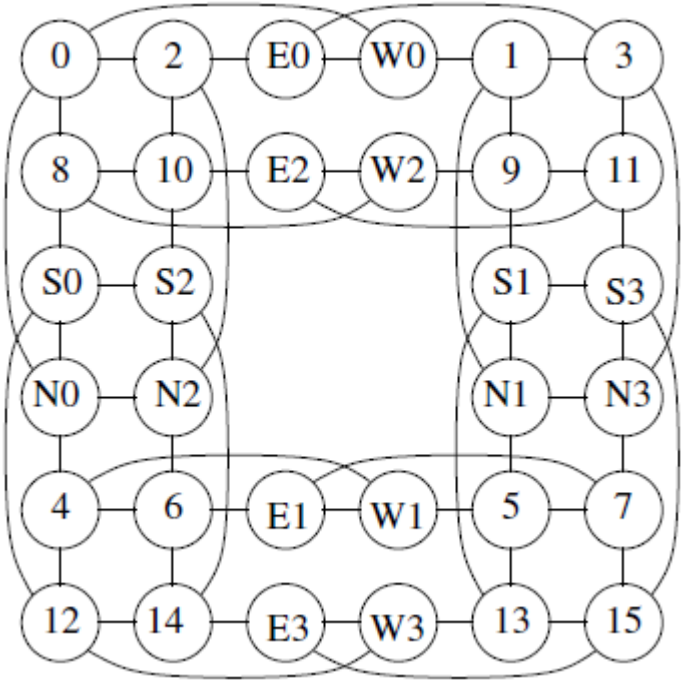
- Assume that endpoints are given for each flow
- Routes must be chosen to avoid deadlock
  - Always an issue on wormhole switching networks.
- Routes must compose a feed-forward network
  - To enable application of the main DNC results
- Select routes for each flow to optimize use of the global network capacity, while guaranteeing a fair bandwidth allocation to the flows
- Max-min fairness: an increase of any flow rate must be at the cost of a decrease of some already smaller flow rate
  - Max-min fair allocation is solved by the simple 'Water Filling' algorithm in case there is one unique path available per flow
  - In case of multiple paths available per flow and splittable flows, the problem is of polynomial time complexity and can be solved as a series of linear programs
  - When only a single path among those available can be assigned to the flow, the max-min fairness with unsplittable path problem becomes NP-hard

Deadlock-Free Deterministic Routing

- Deadlock results from circuits of **agents** and **resources** connected by a **wait-for** relation [Dally & Seitz 1987]
  - Circuit switching: agents are connections; resources are channels
  - Wormhole switching: agents are packets; resources are link buffers
    - Links between routers and links internal to routers ('turns')
- Resource dependence graph
  - Whenever an agent is holding resource R_i while waiting for resource R_j, a dependence between R_i and R_j exists
  - Deadlock can be avoided by eliminating circuits in dependence graph
- Deadlock-free packet switching
  - Restrict routing to remove circuits from the resource dependence graph
  - Equivalently, there must be a numbering of the resources such that each allowed route traverses increasingly numbered resources

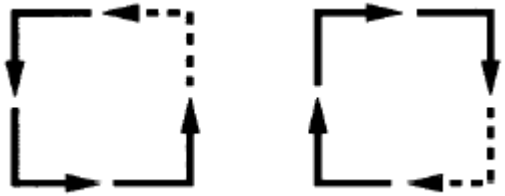
Deadlock-Free Routing Wormhole Switching

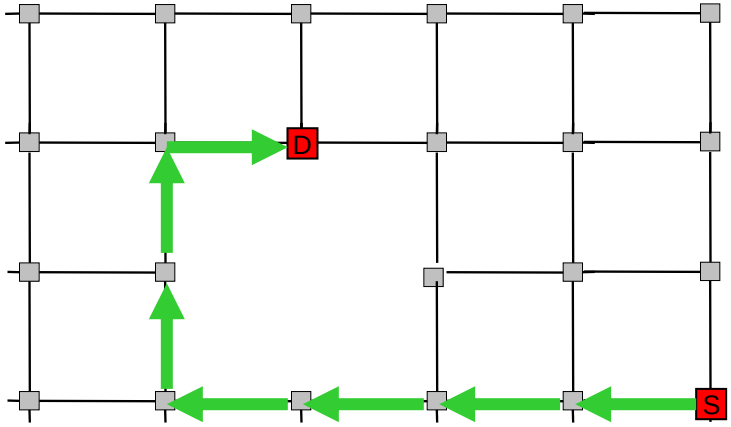
- Deadlock-free wormhole routing techniques when the network topology is a 2D mesh
  - Dimension Order (X-Y on 2D meshes)
  - Turn Model [Glass & Ni 1994] (not the same as Turn Prohibition [Starobinski et al. 2003])
  - Odd-Even [Chiu 2000] (better path diversity than Turn Model)
  - Hamiltonian Odd-Even [Bahrebar & Stroobandt 2015] (support of multicasting)
- Application to the MPPA® NoC
  - Isolate a 2D mesh in NoC topology and apply one of these routing techniques



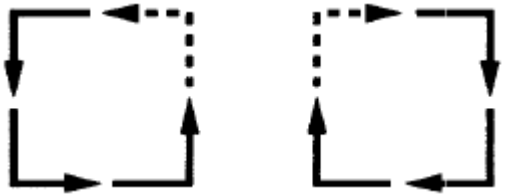
Turn Models

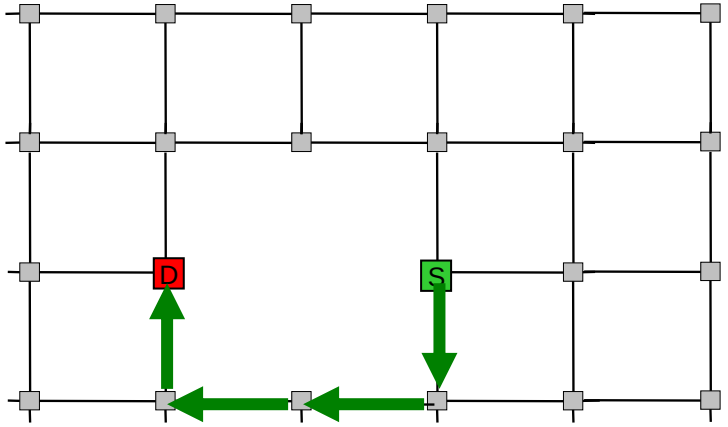
- West First
  - No North-West turn
  - No South-West turn





- North Last
  - No North-West turn
  - No North-East turn



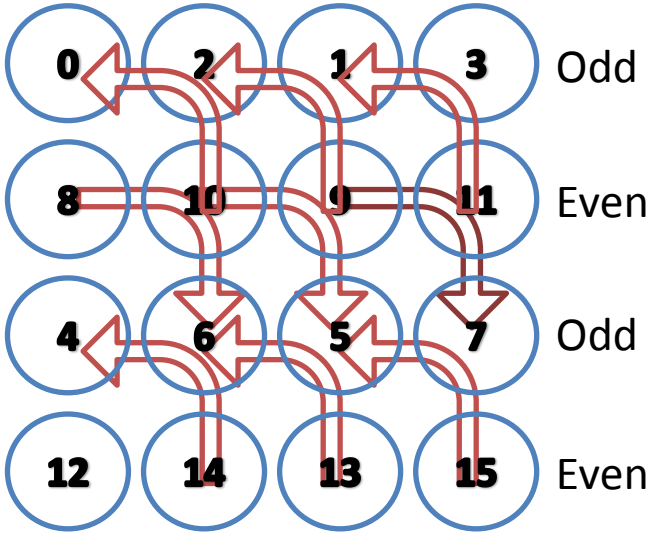


Odd–Even Routing

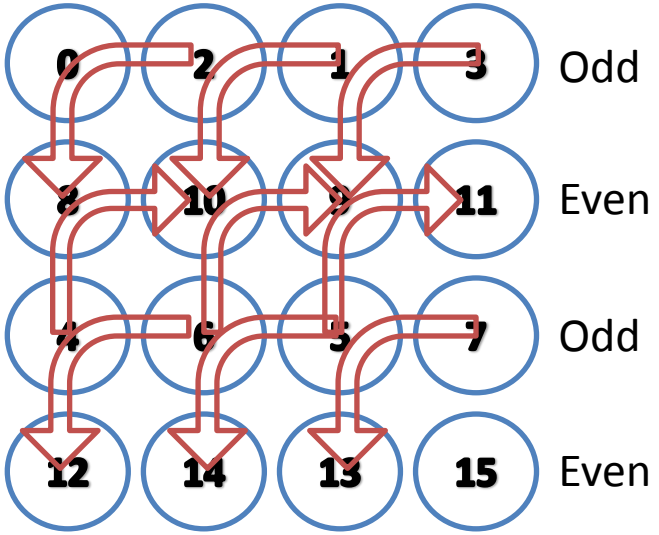
- The adaptiveness of the Turn Model [Glass & Ni 1994] is uneven
  - At least half of the source-destination pairs are restricted to having only one minimal path [Chiu 2000]
- The Odd-Even turn model [Chiu 2000] is fully adaptive
  - Even columns: East-North and East-South turns are prohibited
  - Odd columns: North-West and South-West turns are prohibited
  - 180-degree turns are prohibited
- Hamiltonian-based Odd–Even [Bahrebar & Stroobandt 2015]
  - Designed to be compatible with the Multi-Path (MP) and the Column-Path (CP) routing algorithms for path-based multicast
  - Considers Odd/Even rows instead of Odd/Even columns
  - 180-degree turns are prohibited

Hamiltonian Odd-Even Prohibited Turns

- Even rows
  - East-South turn prohibited
  - North-West turn prohibited



- Odd rows
  - North-East turn prohibited
  - West-South turn prohibited

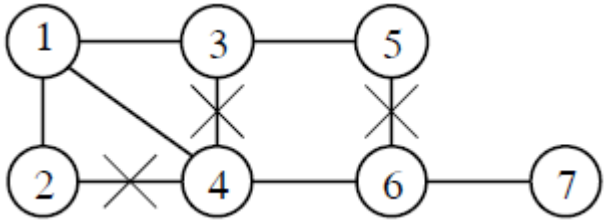


Feed-Forward Flow Routing

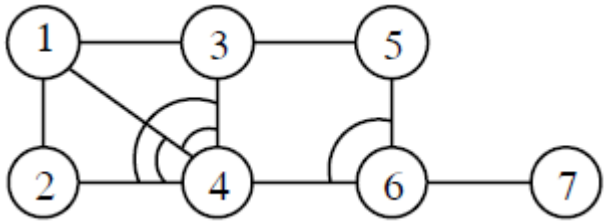
- A network is feed-forward if it is possible to find a numbering of its links such that for any flow through the network, the numbering of its traversed links is an increasing sequence
- Application of a deterministic deadlock-free routing algorithm for wormhole switching ensures that the flows are feed-forward
  - The links considered in a feed-forward network form a subset of the resources considered for deadlock in wormhole switching
  - The numbering of these resources so that they are allocated in ascending order by any packet is also a numbering of the links which is traversed in ascending order by the flows
- Conversely, feed-forward flow routing on a wormhole switching network implies deadlock-free routing of flow packets
  - Feed-forward flow routing ensures that the Turnnet has no cycles
  - With wormhole switching, acyclicTurnnet implies acyclic resource graph

Feed-Forward Routing Techniques

- Spanning tree routing
  - Construct a spanning tree of the network graph and prohibit use of links outside the spanning tree



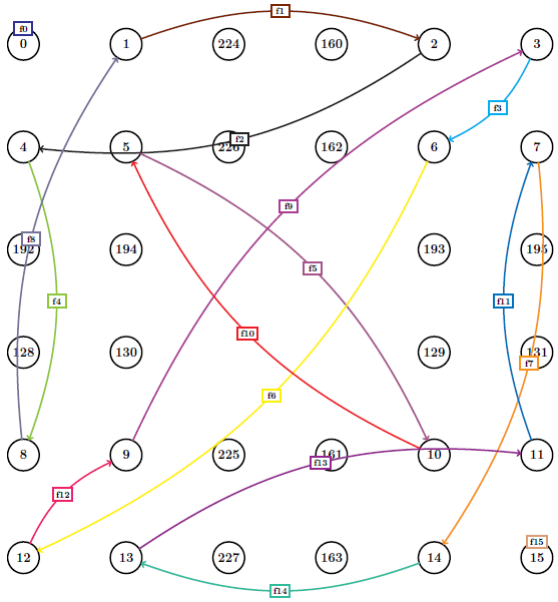
- Up-Down routing
  - Construct a spanning tree of the network graph, order nodes according to their tree level, and prohibit turns (a,b,c) such that a < b and b > c

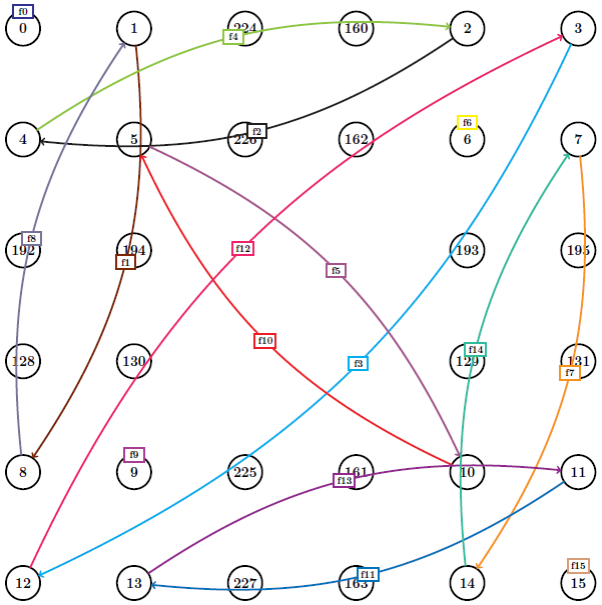


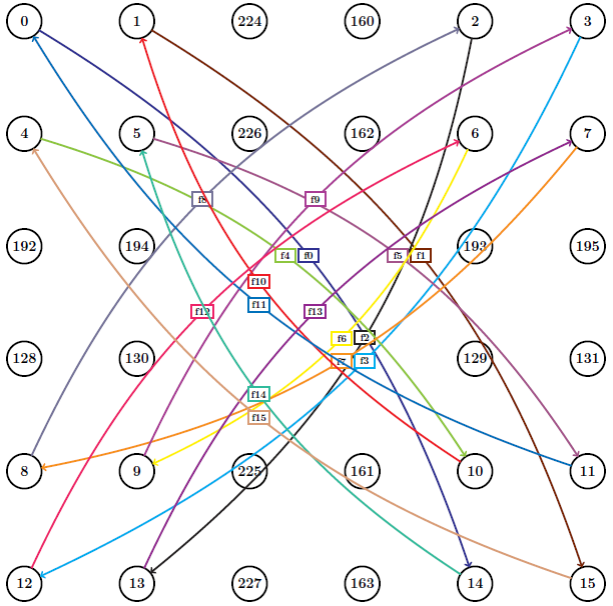
- Turn Prohibition [Starobinski et al. 2003]
  - Recursively break all the link cycles while preserving global connectivity
- Simple Cycle Breaking [Levitin et al. 2010]
  - Improvement of Turn Prohibition, still assume bi-directional turns

Experimental Comparison Setup

- Compare deadlock-free routing and feed-forward routing algorithms on flow routing problems
  - Example 1 and Example 2 extracted from NoC and AFDX papers
  - Bit-Complement, Bit-Reverse, Shuffle, Tornado on MPPA® NoC

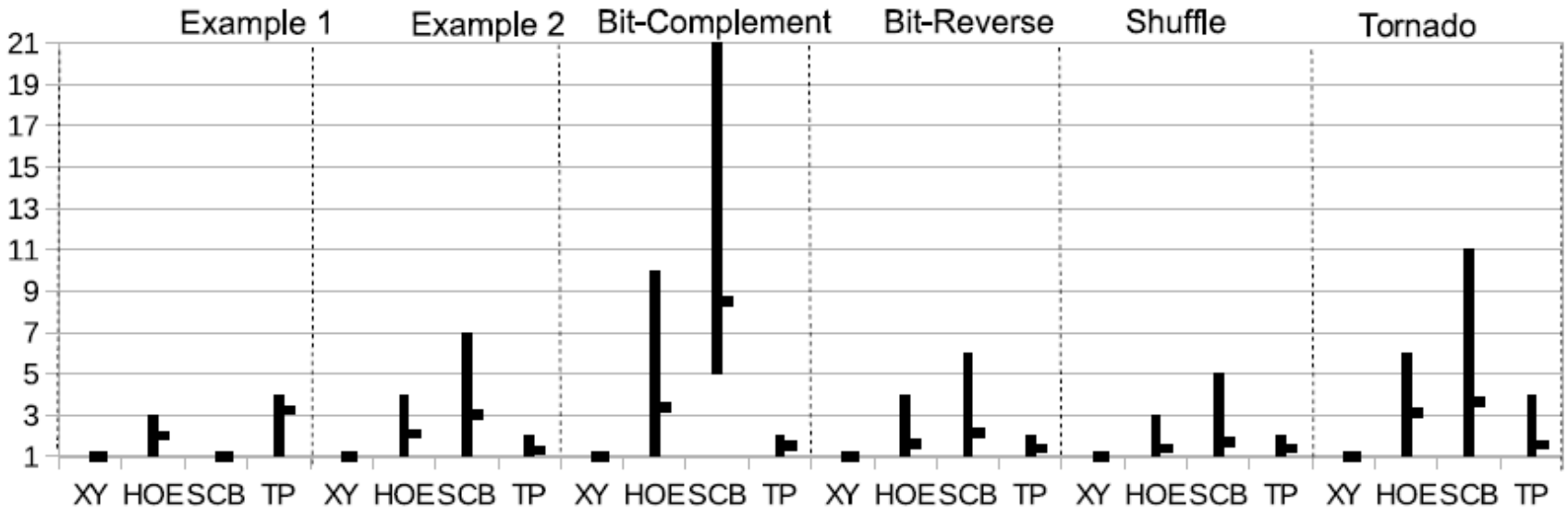






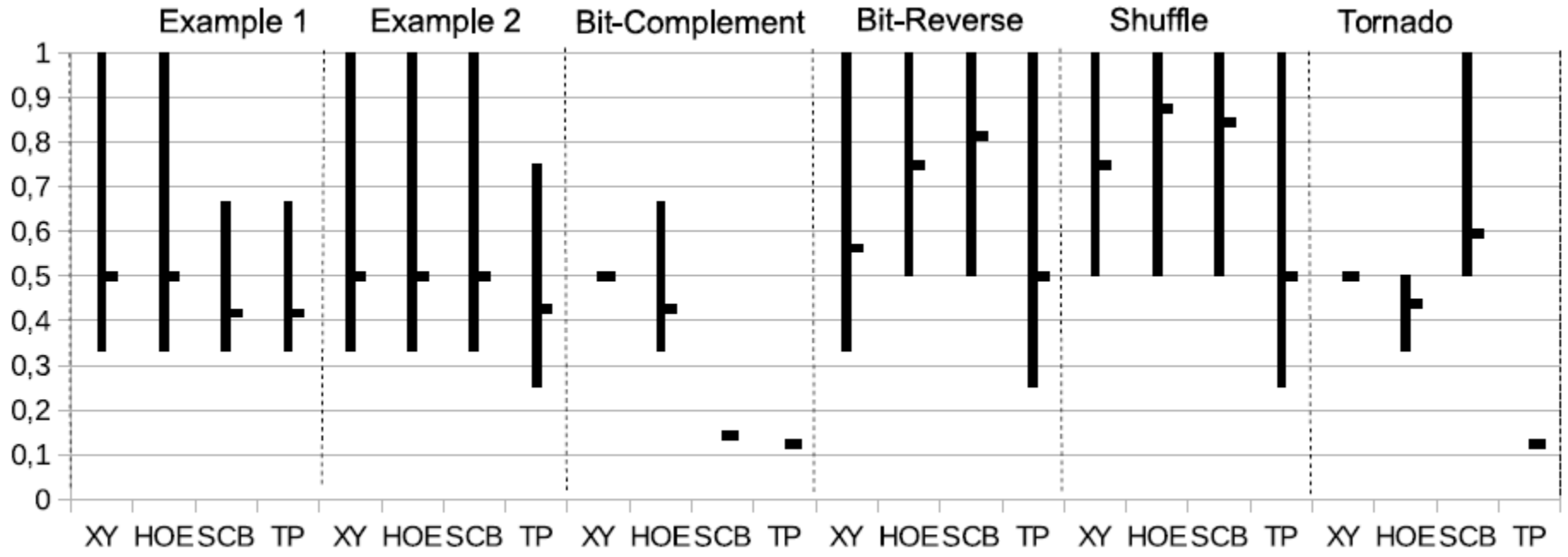
Minimal Path Diversity Results

- For each flow, enumerate all minimal paths between endpoints allowed by the routing algorithm
  - Figure displays min, max and and average of minimal path counts



Max-Min Fair Flow Rates Results

- Solve max-min fair routing with unsplittable paths by enumeration
  - Figure displays rate range and rate average over flows
- On Bit-Complement, X-Y outperforms HOE applied once
  - HOE should be applied to the 2D-grid not only on rows but also on columns

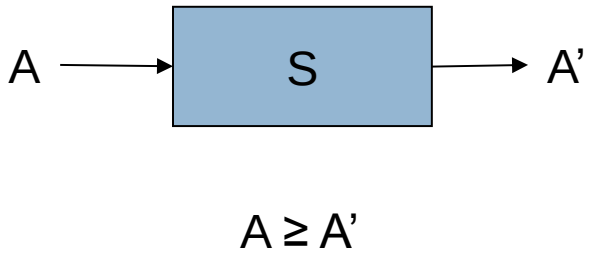


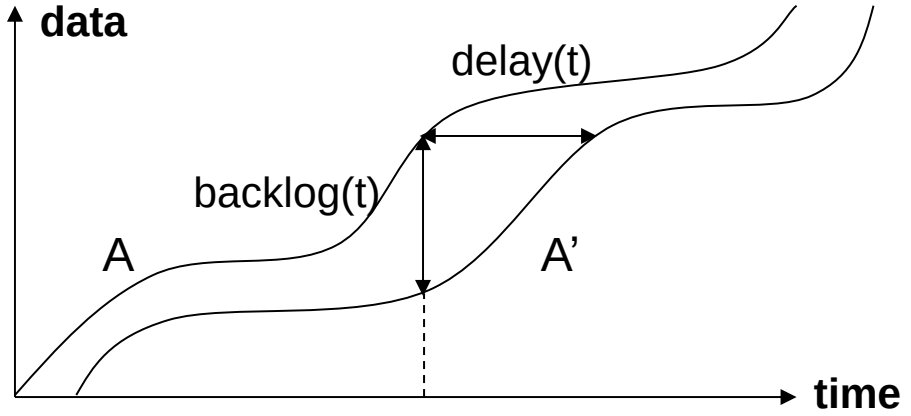
Outline

- MPPA®-256 Bostan Processor
- MPPA® Application Environment
- MPPA® Bostan NoC Architecture
- Max-Min Feed-Forward Routing
- Deterministic Network Calculus (DNC)
- DNC Application to the MPPA® Bostan NoC
- Conclusions

Deterministic Network Calculus

- Compute deterministic upper/lower bounds in communication networks
- Flows are represented by cumulative data transferred up to time t
- Servers are abstracted as relations between input and output flows





- Framework based on (min,+) dioid instead of (+,*) ring or field

$(f \otimes g)(t) = inf_{0 \le s \le t}(f(t-s) + g(s))$ convolution

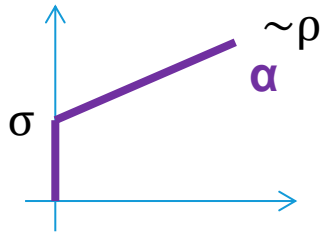
$(f \oslash g)(t) = sup_{s>0}(f(t+s) - g(s))$ deconvolution

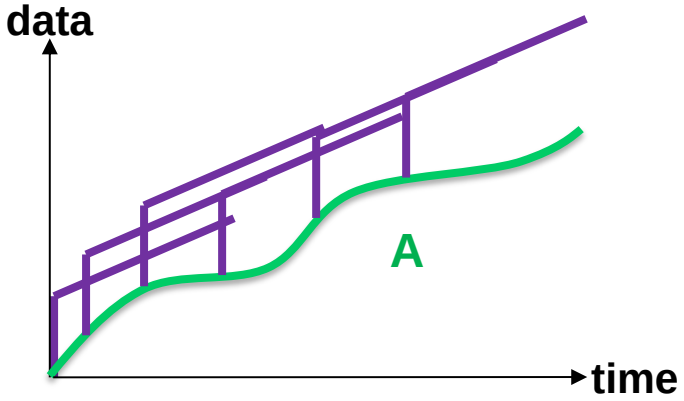
$f \oslash g \le h \Leftrightarrow f \le h \otimes g$ duality

Arrival Curves

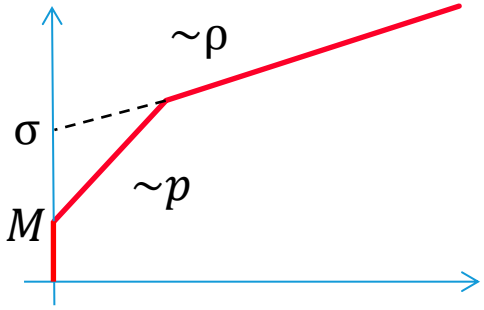
- An arrival curve α(t) is a traffic contract on a cumulative arrival A(t):
  - $\forall t, d \geq 0, A(t+d) - A(t) \leq \alpha(d)$ equivalent to $A \leq A \otimes \alpha$

  - Leaky-bucket arrival curve:
    $\alpha(t) = (\sigma + \rho t)_{1_{t>0}}$



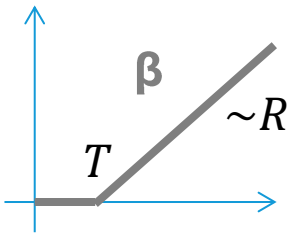


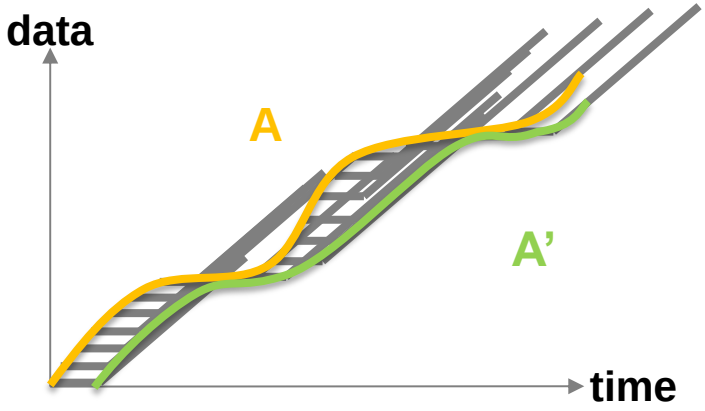
  - TSPEC arrival curve:
    $\alpha(t) = \min(M + pt, \sigma + \rho t)_{1_{t>0}}$



Service Curves

- A server has a lower service curve β(t) iff for any input A(t):
  - Output flow A'(t) satisfies $A' \geq A \otimes \beta$ and $\beta(0) = 0$
  - Rate-latency service curve:
    $\beta(t) = R\,[t - T]_+$

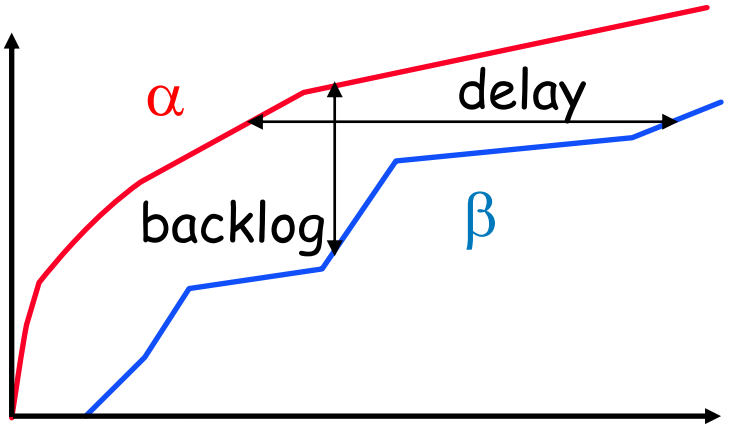




- A server has a strict service curve β(t) iff for any input A(t):
  - For any period (s, t] during which the flow is backlogged
    $A'(t) - A'(s) \geq \beta(t - s)$

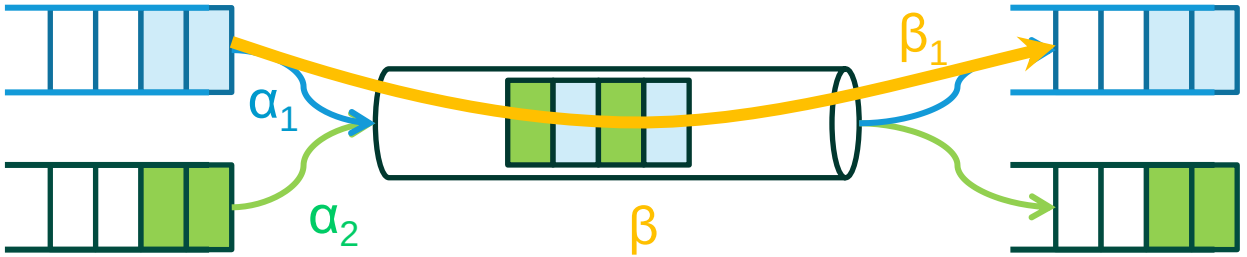
Main DNC Results

- Constraint propagation rule
  - A flow A(t) with arrival curve $\alpha(t)$ that traverses a server with service curve $\beta(t)$ results in a flow A'(t) constrained by arrival curve $\alpha \oslash \beta(t)$
- Tandem composition rule
  - The service curve of a tandem of two of servers with respective service curves $\beta_1(t)$ and $\beta_2(t)$ is the convolution $\beta_1 \otimes \beta_2(t)$
- Tight delay and backlog bounds
  - If a flow has arrival curve $\alpha(t)$ and a server offers service curve $\beta(t)$:
    - backlog = $\max_{t \geq 0} (\alpha(t) - \beta(t))$
    - delay = $h(\alpha, \beta) = \max_{t \geq 0} \{ \inf\ s \geq 0 : \alpha(t) \leq \beta(t+s) \}$



Flow Aggregation in Servers

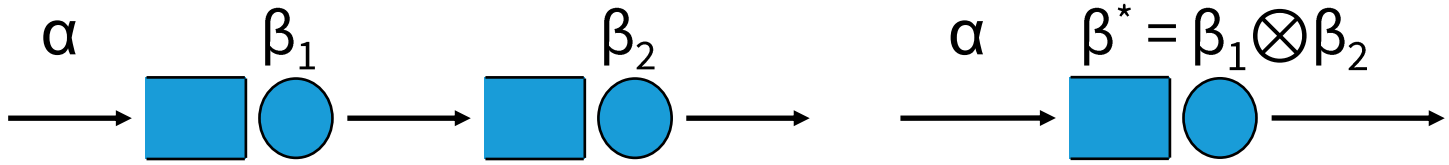
- Blind multiplexing (packets of different flows served in arbitrary order)
  - Assume a node serving the aggregate of two flows with the **strict** service curve β(t); assume flow 2 is α_2-smooth
  - Then a service curve for flow 1 is $\beta_1(t) = [\beta(t) - \alpha_2(t)]^+$



- FIFO multiplexing (packets of different flows in same FIFO queue)
  - Assume a node serving the aggregate of two flows in FIFO order with the **lower** service curve β(t); assume flow 2 is α_2-smooth; define the β_θ^1 family as $\beta_\theta^1(t) = [\beta(t) - \alpha_2(t - \theta)]^+ 1_{\{t>\theta\}}$
  - For θ≥ 0, if β_θ^1 is wide-sense increasing, it is a service curve for flow 1

End-to-End Delay Bounds

- Without aggregation: use tandem composition (PBOO)
  - Delay = h(α, β*) α the arrival and β* the convolution of service curves



- With aggregation [Bouillard & Stea 2015]:
  - Separated-Flow Analysis (SFA)
    - First compute the equivalent service curves for tagged flow on its path
    - Then compute the convolution of the curves thus obtained
  - Pay Multiplexing Only Once (PMOO)
    - Isolate path segments carrying the same flows
    - First compute the convolution of the service curves on the segments
    - Then compute the equivalent service for tagged flow
  - Neither method is tight or best, however the SFA is simpler to engineer

Outline

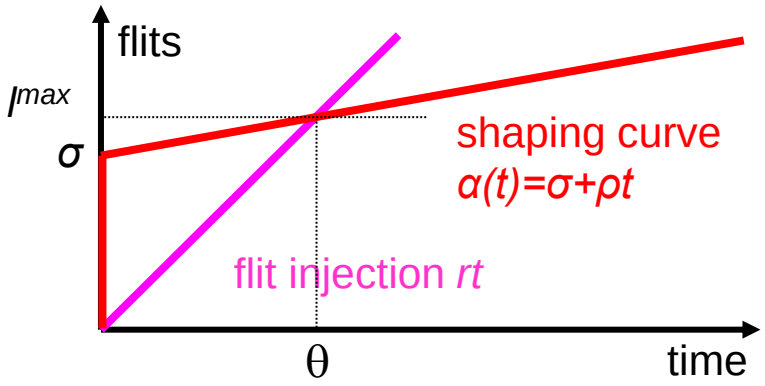
- MPPA®-256 Bostan Processor
- MPPA® Application Environment
- MPPA® Bostan NoC Architecture
- Max-Min Feed-Forward Routing
- Deterministic Network Calculus (DNC)
- DNC Application to the MPPA® Bostan NoC
- Conclusions

MPPA® NoC Guaranteed Services

- Assume tasks are allocated to endpoints (NoC nodes)
- Allocate one path for each pair of endpoints with deadlock-free routing
  - On wormhole switching NoC, this ensures feed-forward flows
    - Hamiltonian Odd-Even routing on 2D grid sub-topology of NoC
  - Solve max-min fairness with unsplittable path constraint (NP-hard)
    - This computes the maximum flow rates $\overline{\rho}$ between each endpoint
- Compute upper bounds on burstiness (σ) at ingress for each flow
  - Turn queues must not overflow, which constrains the initial burstiness (σ)
  - Compute service curve offered to turn queues using either round-robin packet scheduling or blind multiplexing
  - Use FIFO multiplexing formulas for aggregation inside turn queues
    - For the burstiness increase and for the left-over service curves
- Compute the end-to-end delay bounds for each flow using SFA

Packet Injection Constraints with Link Shaping

- Arrival curve is $\gamma_{\rho,\sigma}$ shaped by the 'turn' link at rate $r \geq \rho$
  - Let r be the injection rate and l^{max} the maximum packet size
  - $r\tau_i = \sigma_i + \rho_i\tau_i \Leftrightarrow \tau_i = \frac{\sigma_i}{r-\rho_i}$
  - $\sigma_i \geq l^{max} - \rho_i\tau_i \Leftrightarrow \sigma_i \geq l^{max} - \rho_i \frac{\sigma_i}{r-\rho_i} \Leftrightarrow \sigma_i \geq l^{max}\frac{r-\rho_i}{r}$



Turn Queue Service Curves

- On the MPPA® NoC, the output link arbiters operate in round-robin on turn queues at the packet granularity, while each queue contains flows aggregated in FIFO
- The service offered to each queue of a link arbiter abstracted as $\beta_{R,T}$
  - Either, the rate and latency ensured by round-robin packet scheduling

$$R^j = \frac{r l_{F^j}^{\min}}{l_{F^j}^{\min} + \sum_{k \in B^j} l_{F^k}^{\max}} \text{ and } T^j = \frac{\sum_{k \in B^j} l_{F^k}^{\max}}{r}$$

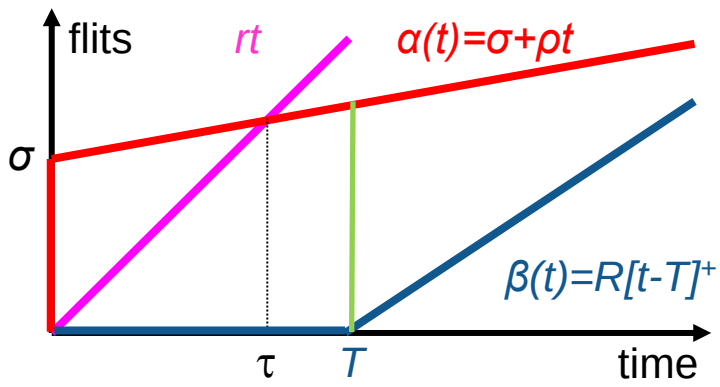
  - Or, the residual service guaranteed by blind multiplexing across queues when the round-robin service does not apply

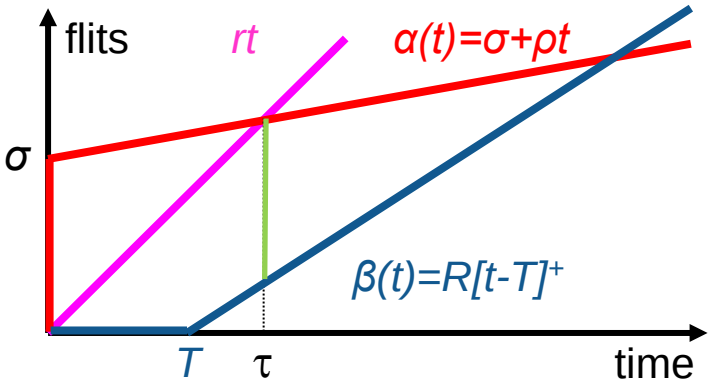
$$R^j = r - \sum_{k \in B^j} \rho^k \text{ and } T^j = \frac{\sum_{k \in B^j} \sigma^k}{r - \sum_{k \in B^j} \rho^k}$$

- See paper for the explanation of these $\beta_{R,T}$ formulas

Turn Queue Constraints with Link Shaping

- Service curve to queue is abstracted as $\beta_{R,T}$
  - Let $\gamma_{\rho,\sigma}$ be the arrival curve of the flow aggregate in queue
- Backlog in the turn queue is $b = \max_{t\geq 0}(\alpha(t) - \beta(t)) \leq q_{size}$
  - If $\tau \leq T \Leftrightarrow \sigma \leq (r-\rho)T$ then $b = \sigma + \rho T$
  - Else $b = r\tau - R(\tau - T) \Leftrightarrow b = \frac{(r-R)}{(r-\rho)}\sigma + RT$
  - Always safe to use formula $b = \sigma + \rho T$

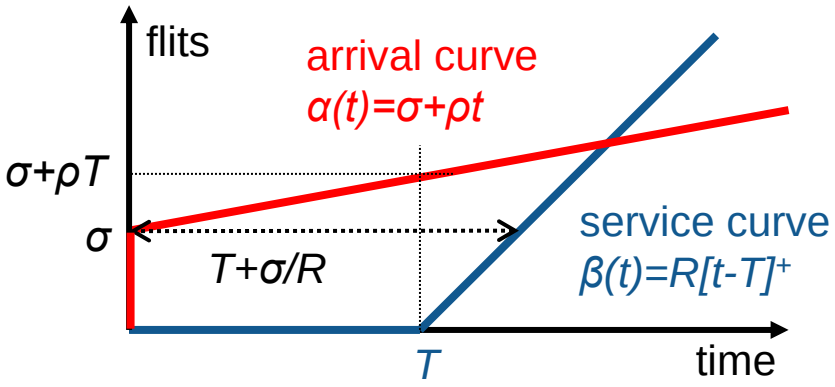




Burstiness Increase of Flows

- Service curve to queue is abstracted as $\beta_{R,T}$
  - Let $\gamma_{\rho,\sigma}$ be the arrival curve of the flow in queue

- No multiplexing in queue
  - $(\sigma_i, \rho_i) \rightarrow (\sigma_i + \rho_i T\ , \rho_i)$



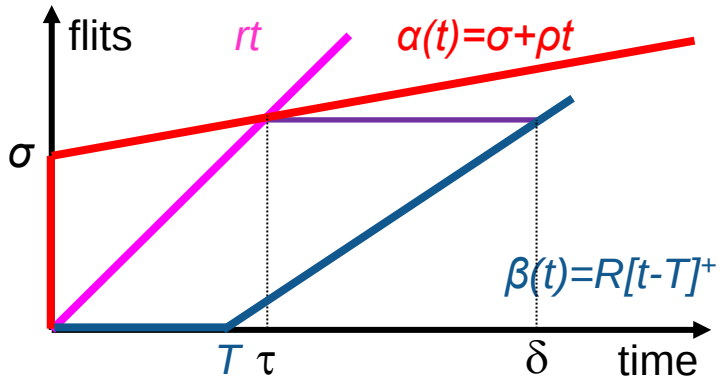
- FIFO multiplexing in queue
  - Let $\gamma_{\rho',\sigma'}$ be the sum of arrival curves of other flows in the queue
  - $(\sigma_i, \rho_i) \rightarrow (\sigma_i + \rho_i(T + \frac{\sigma'}{R}), \rho_i)$

FIFO Multiplexing with Link Shaping

- Burstiness increase due to FIFO multiplexing, general case
  - Assume that flow 1 is constrained by γ_{ρ_1,σ_1} and flow 2 is constrained by a sub-additive arrival curve α_2. Assume that the node guarantees to the aggregate of the two flows a rate latency service curve $\beta_{R,T}$. Call $\rho_2 = \inf_{t>0} \alpha_2/t$ the maximum sustainable rate for flow 2.
  - If $\rho_1 + \rho_2 < R$, then at the output, flow 1 is constrained by γ_{ρ_1,b_1}
  - $b_1 = \sigma_1 + \rho_1(T + \frac{B}{R})$ with $B = \sup_{t\geq 0}[\alpha_2(t) + \rho_1 t - Rt]$
- With link shaping, $\alpha_2(t) = \min(rt, \rho_2 t + \sigma_2)$ and $\tau = \frac{\sigma_2}{r-\rho_2}$
  - $B = \sup(\sup_{0\leq t\leq \tau}[rt + \rho_1 t - Rt], \sup_{t\geq\tau}[\rho_2 t + \sigma_2 + \rho_1 t - Rt]) = \frac{r+\rho_1-R}{r-\rho_2}\sigma_2$
  - $b_1 = \sigma_1 + \rho_1(T + \frac{\sigma_2(r+\rho_1-R)}{R(r-\rho_2)})$ with $\frac{\sigma_2(r+\rho_1-R)}{R(r-\rho_2)} < \frac{\sigma_2}{R}$ as $\rho_1 + \rho_2 < R$

End-to-End Latency with Link Shaping

- End-to-end residual service curve for flow of interest is $\beta_{R,T}$
  - With R the min of the residual rates and T the sum of the residual latencies
- Maximum delay is $d = \max_{t \geq 0} \{ \inf\ s \geq 0 : \alpha(t) \leq \beta(t+s) \}$
  - Maximum horizontal deviation is reached between τ and δ
  - $r\tau = (\delta - T)R$ with $\tau = \frac{\sigma}{r-\rho}$
  - $d = \delta - \tau = T + \frac{\sigma(r-R)}{R(r-\rho)}$



Outline

- MPPA®-256 Bostan Processor
- MPPA® Application Environment
- MPPA® Bostan NoC Architecture
- Max-Min Feed-Forward Routing
- Deterministic Network Calculus (DNC)
- DNC Application to the MPPA® Bostan NoC
- Conclusions

Lessons Learned

- Deterministic deadlock-free routing and feed-forward flow routing are equivalent on a wormhole switching NoC
  - First application of Turn Prohibition and Simple Cycle Breaking to a NoC
- Computing (maximum) flow rates in a preliminary step enables the formulation of DNC equations to be linear with the burstiness variables
  - Thanks to feed-forward flows, set of acyclic inequalities solved in one pass
- Good solutions to the max-min fairness with unsplittable paths problem instances on MPPA® NoC can be found by heuristic
  - Round solutions to the max-min fairness with splittable paths problem and enumerate over splitted paths with the same maximal sub-flow rate
- Hamiltonian Odd-Even routing seems to perform best on MPPA® NoC
  - Must be applied vertically and horizontally on a 2D-grid topology
  - Motivates a NoC with two virtual channels on the MPPA® Coolidge

Thank you

KALRAY S.A.
Paris - France

86 rue de Paris,
91 400 Orsay
France

Tel: +33 (0) 184 00 00 45
email: info@kalray.eu

KALRAY S.A.
Grenoble - France

445 rue Lavoisier,
38 330 Montbonnot
France

Tel: +33 (0)4 76 18 09 18
email: info@kalray.eu

KALRAY INC.
Los Altos - USA

4962 El Camino Real
Los Altos, CA
USA

Tel: +1 (650) 469 3729
email: info@kalrayinc.com