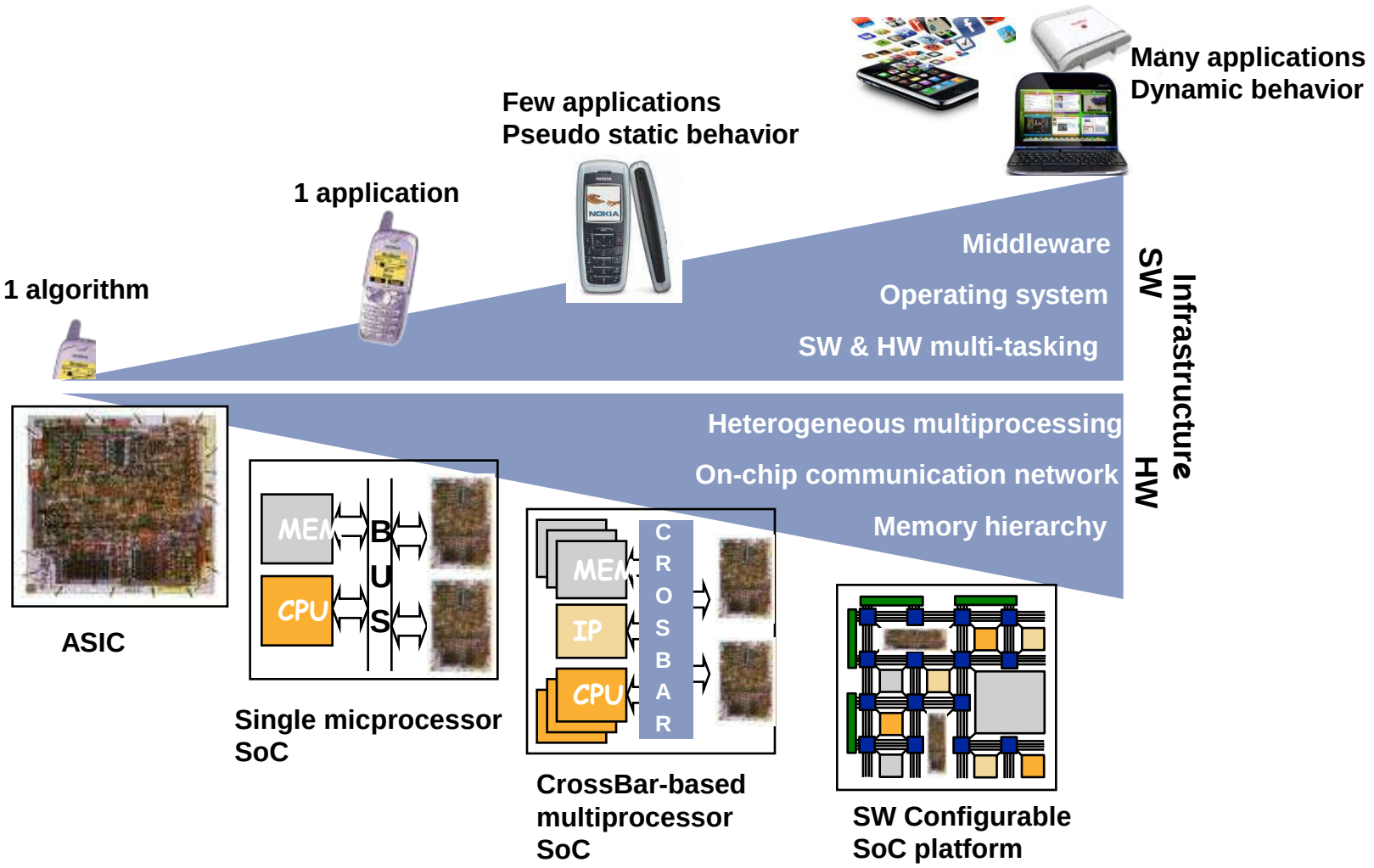
Spidergon STNoC:
The technology that adds value to your System

Marcello Coppola

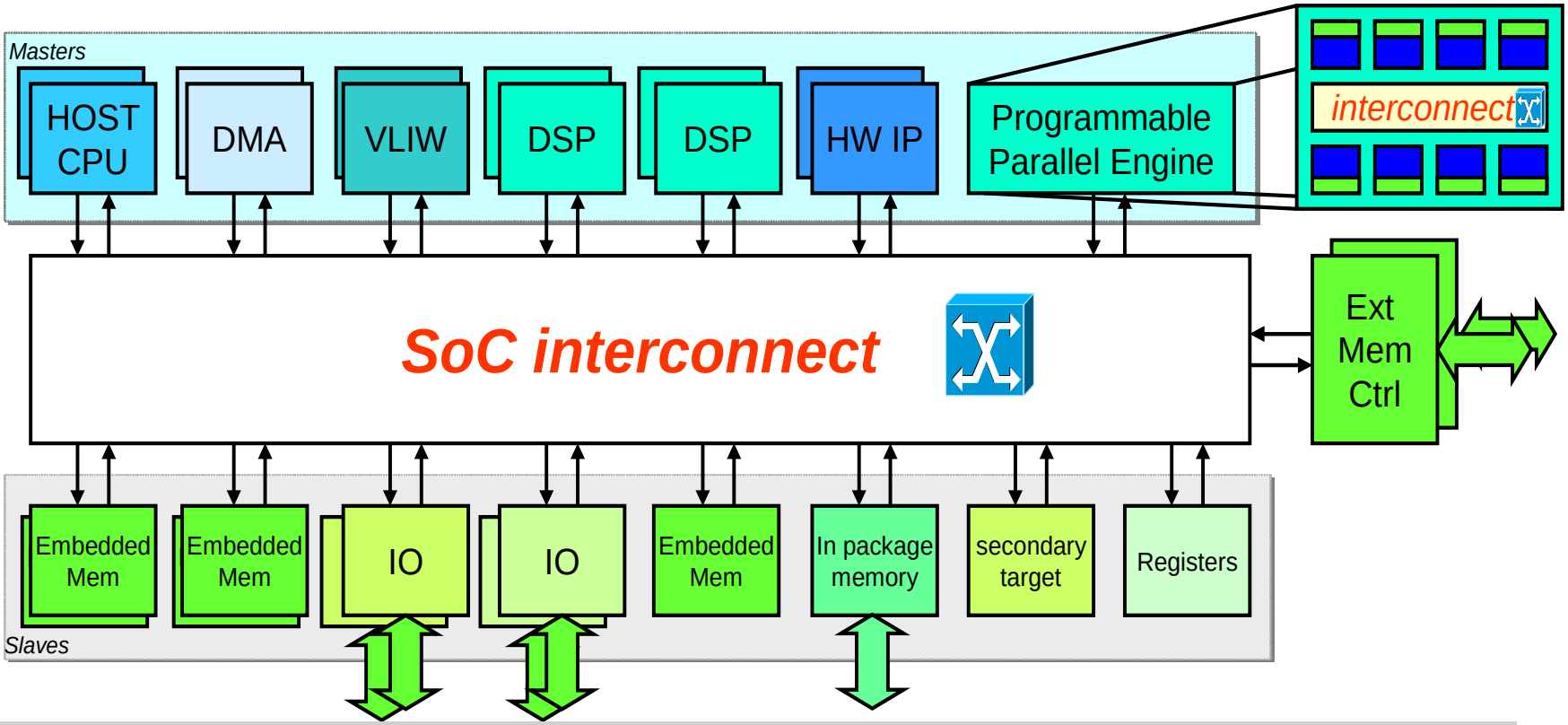
Agenda

- Setting the stage
- Overview on Spidergon STNoC
- Spidergon STNoC backbone
- User experience
- Communication Primitives & Services
- Spidergon STNoC today
- Conclusions

Moore’s law in real life



Multicore SoC architecture: 2004 vision



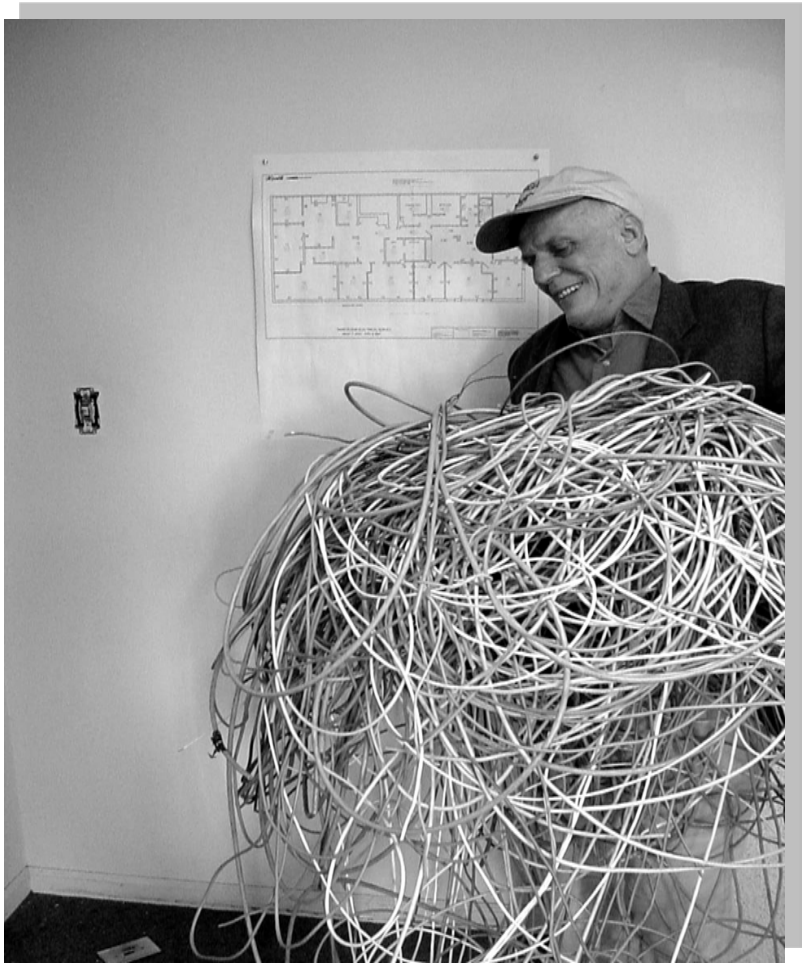
towards…

Open Configurable Heterogeneous Multicore Platforms

Agenda

- Setting the stage
- Overview on Spidergon STNoC
- Spidergon STNoC backbone
- User experience
- Communication Primitives & Services
- Spidergon STNoC today
- Conclusions

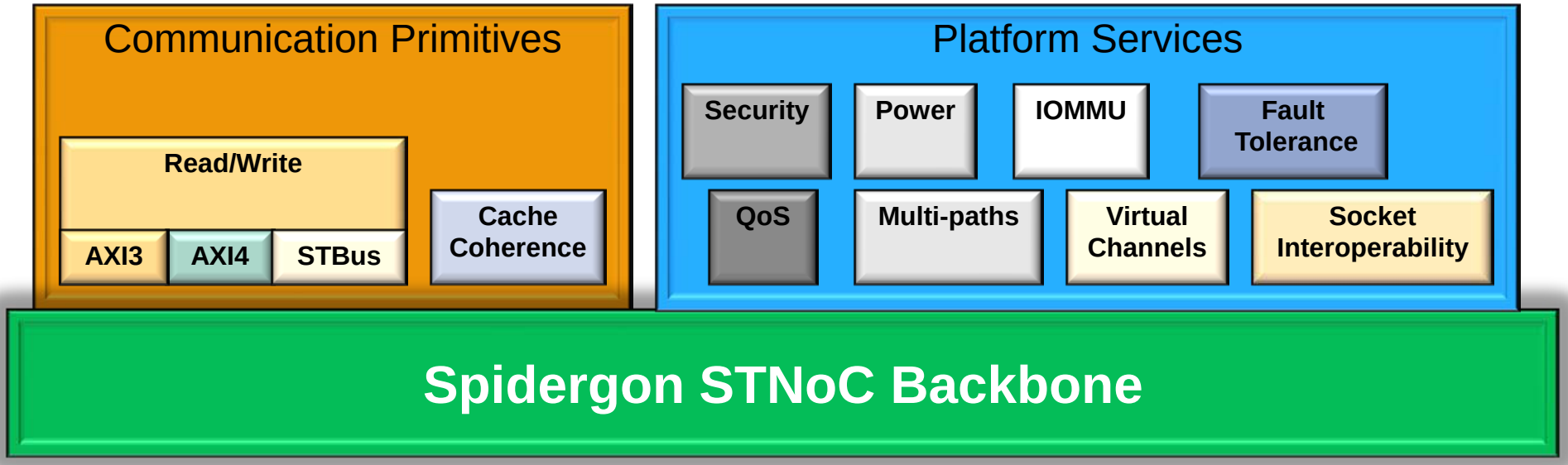
Spidergon STNoC at a glance



- Best in class for multi-protocol support (AMBA, STBUS, custom)
- Solving wire congestion and time closure problems without any area penalty
- Reduce SoC NRE
- Several Innovative Services
- Configurable end extensible technology
- Fully integrated with state-of-the-art verification environment
- Allow post silicon re-configurability by software

Spidergon STNoC technology

- Spidergon STNoC is set of *Communication Primitives* and *Services implemented* on top of a distributed on-chip network (NoC backbone) that connects several components.



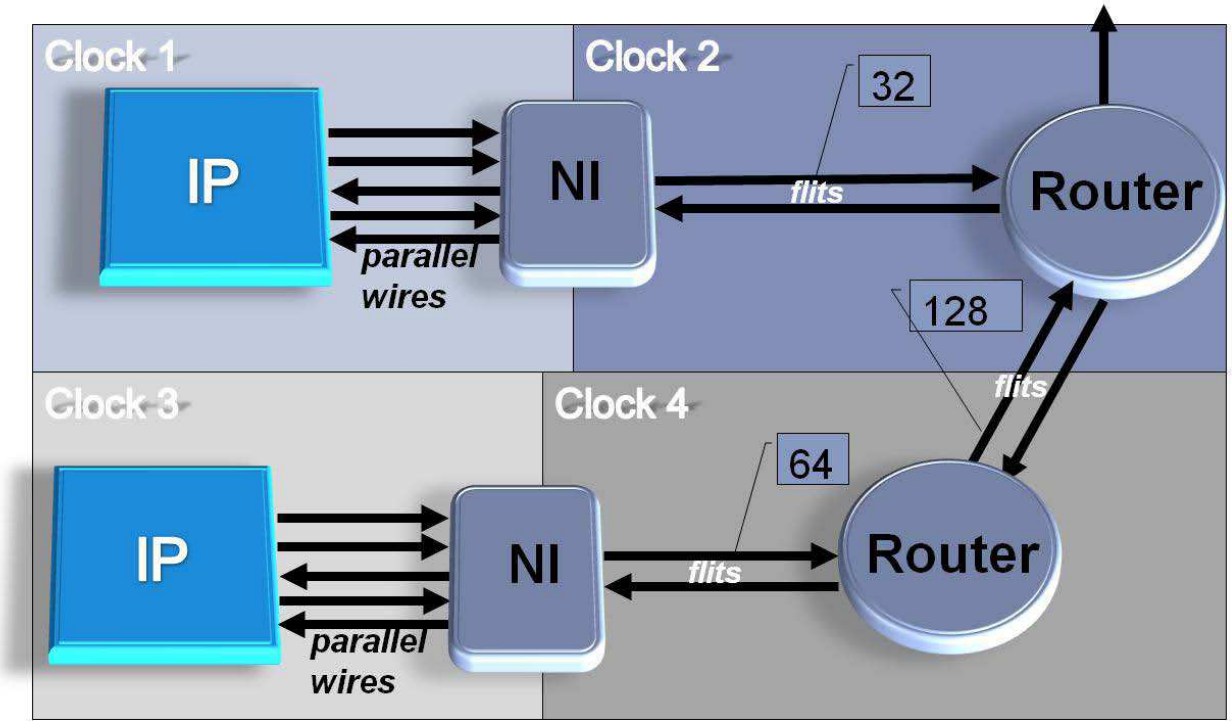
Agenda

- Setting the stage
- Overview on Spidergon STNoC
- Spidergon STNoC backbone
- User experience
- Communication Primitives & Services
- Spidergon STNoC today
- Conclusions

Spidergon STNoC backbone

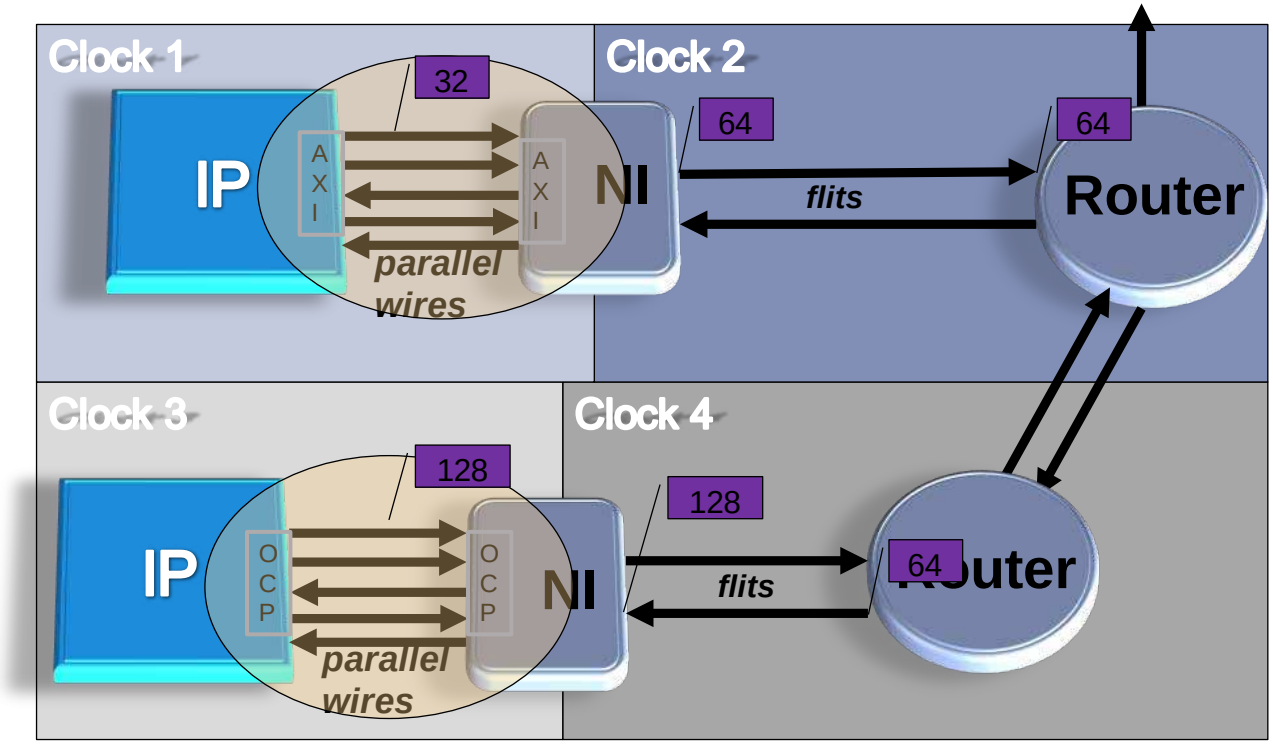
The on-chip network is based on 3 well-defined configurable components called:

- Network Interface
- Router
- Physical link

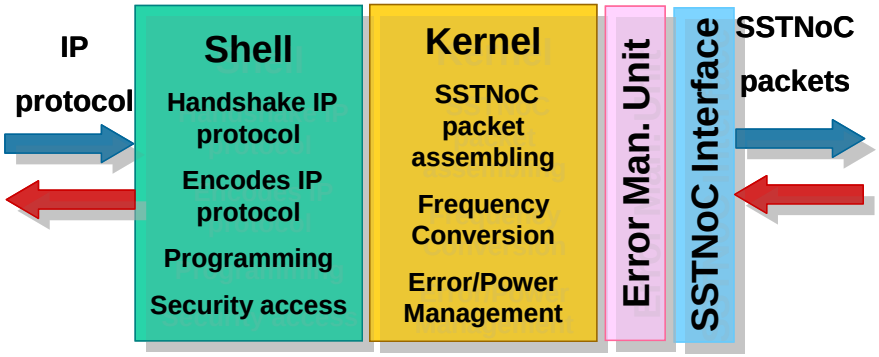


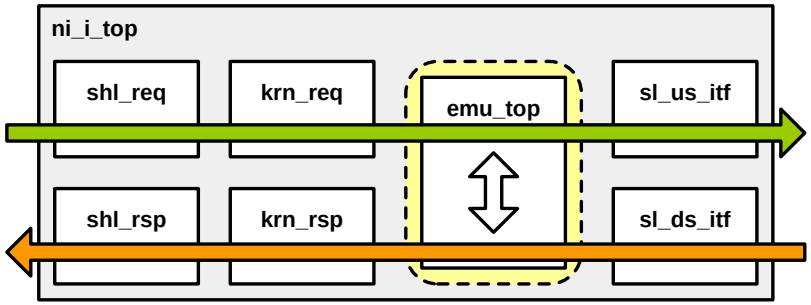
Communication Primitives

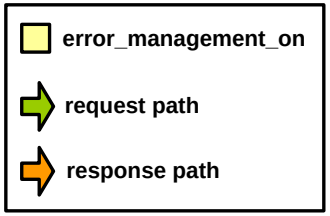
- Natively support (without bridges) different socket protocols (AXI, STBUS, OCP, custom, etc.)
- Efficient upsize and downsize conversion



Network Interface







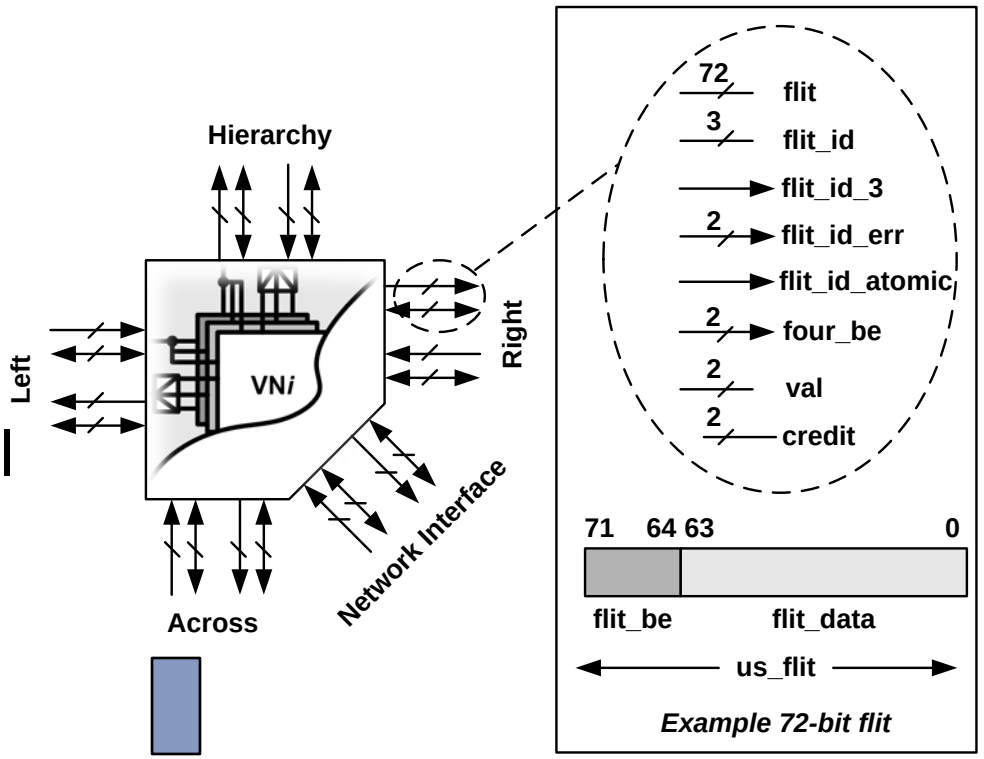
Network Interface Features

- Data bus / flit Size Conversion
  - Upsize
  - Downsize
  - No conversion
- Frequency Conversion
  - Gray counters (any even number of FIFO locations)
  - 2 to 4 Synch. Flip Flops
- 0 to 3 cycles configurable crossing latency
- Store & Forward
  - IP Prot to S-STNoC (packet, chunk, msg)
  - S-STNoC to IP prot (S-STNoC packet)

- Error Management
- Security Management
- Transaction ordering support
- Single/Multiple Target QoS support
- Programming Unit
  - Routing (Dir1, Dir2, DestID fields)
  - QoS FBA fields
  - Security
  - STBus T1 or APB interface

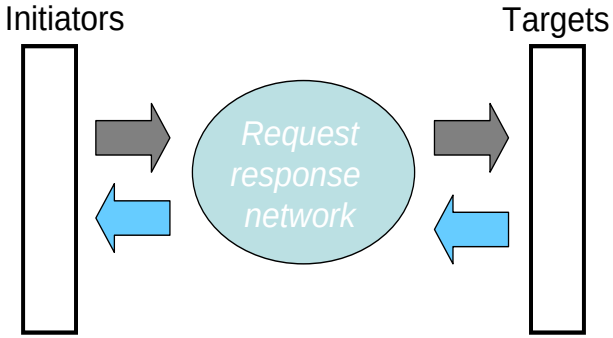
Spidergon Router

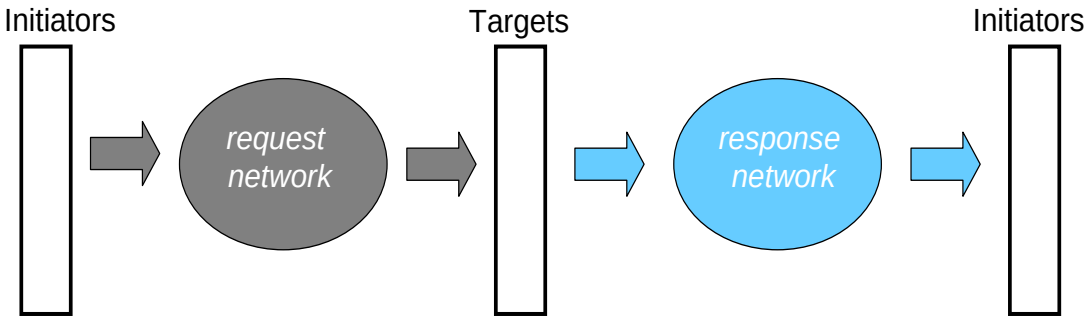
- The **Router** implements the network and the data link layers of the NoC protocol, offering best effort and possible quality of service (QoS) in terms of both latency and throughput
- It is responsible for the transmission of the flits
- The router supports 2 Virtual Channels



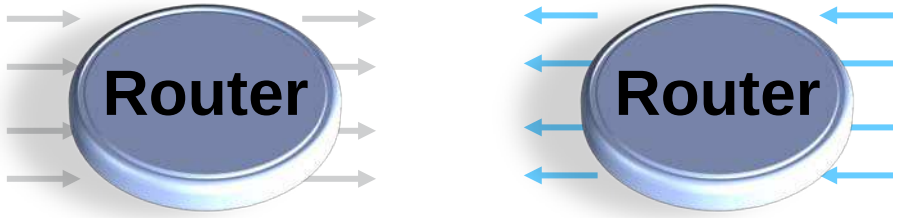
Spidergon STNoC request/response networks

Spidergon STNoC technology allows you to design separated request/ response networks





request & response switch



Less wires per router

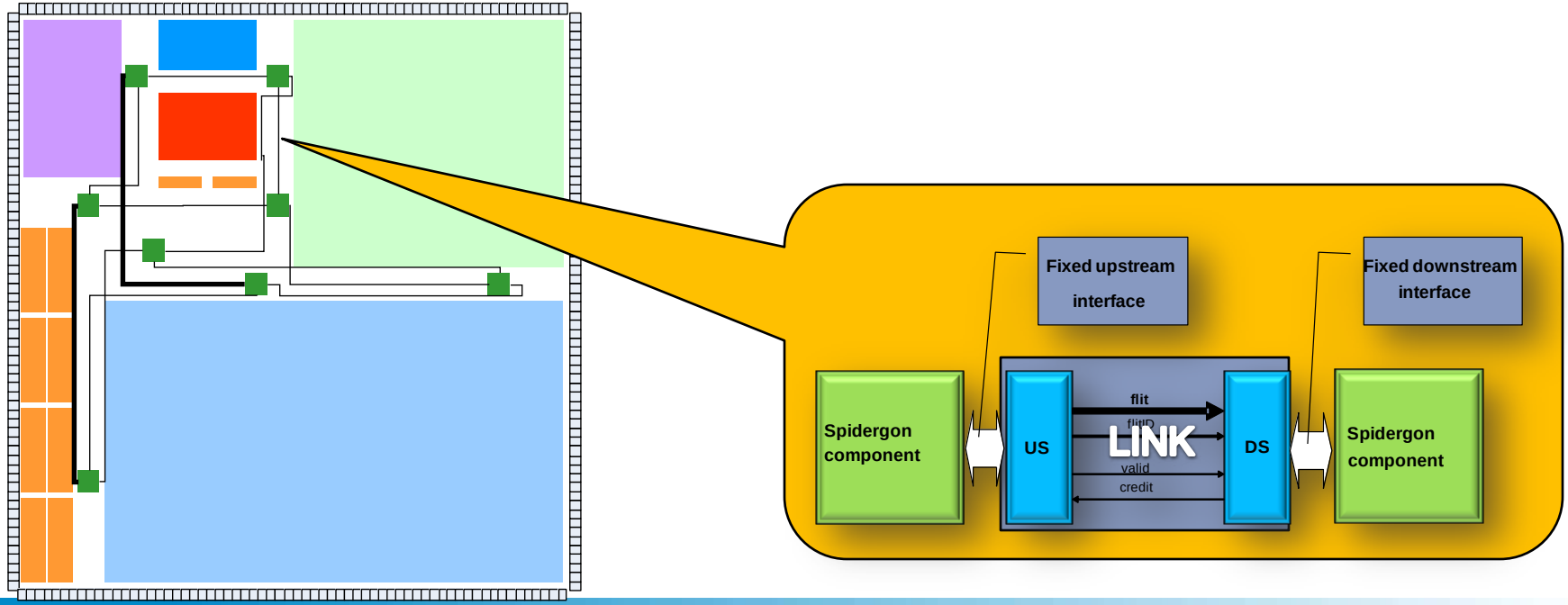
Router Features

- Each Link (L, R, A, NI, H), Port (DS, US), VN (VN1, VN2) is individually instantiable and configurable
- Back routing support
- Configurable flit size: flit payload size16/18, 32/36, 64/72, 128/144 bits, plus flit extra bits
- Optional registers on credits
- IB with/without bypass capability and Optional retiming stage: configurable crossing latency (1 cycle or 2 cycles)

- Configurable Output Queue size: 0 – 64 flits
  - When zero, OQ is not instantiated
- Configurable number of Output Queues to NI
- QoS Support
- Configurable basic arbitration scheme within a faction: LRU, RR, pk_priority
- Configurable virtual channel arbitration scheme: LRU/RR, VN_priority (w/ & w/o lock packet)
- Routing Unit support for hierarchical networks: H link is a gateway between 2 S-STNoC sub-networks
- Routing Unit support for Spidergon Routing or Source Routing mechanisms

Physical Link

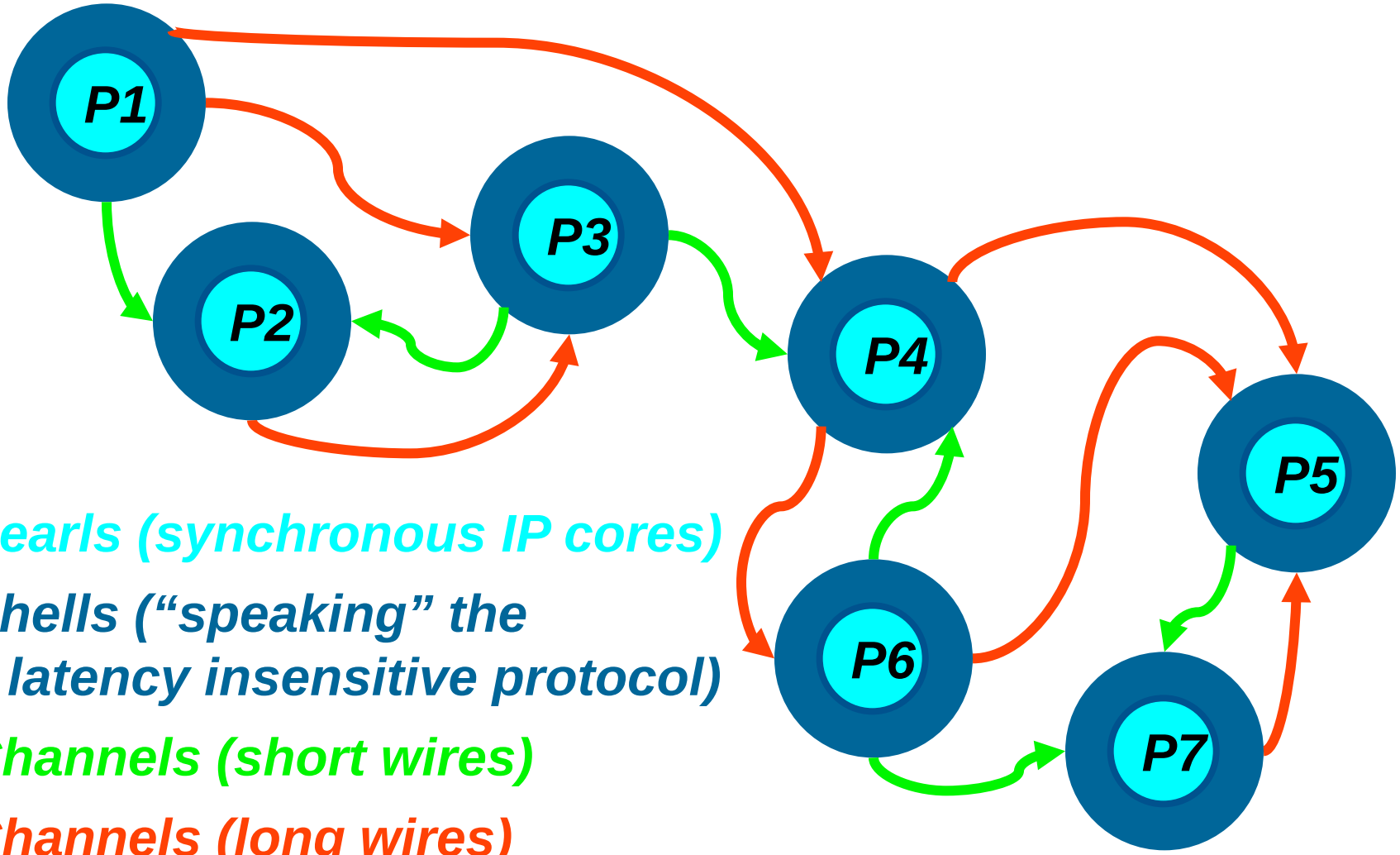
- Composed by a set of wires, an upstream logic and a downstream logic
- Enable efficient physical implementation
- Synchronous version, Adaptive version
- Optimized flow control with virtual channel support



GALS: some previous works

1. Pausible Clocking
   - Yun (ICCAD-96); Villiger et al./ETH (ASYNC-03)
2. Latency-Insensitive Systems
   - Carloni, Sangiovanni-Vincentelli et al. (ICCAD-99);
3. Static Scheduling
   - Casu, Macchiarulo (DAC-04)
3. Fine-Grain Synchronous Handshaking
   - Jacobsen et al./IBM (ASYNC-02)

Latency Insensitive Design (LTI) [ICCAD99]



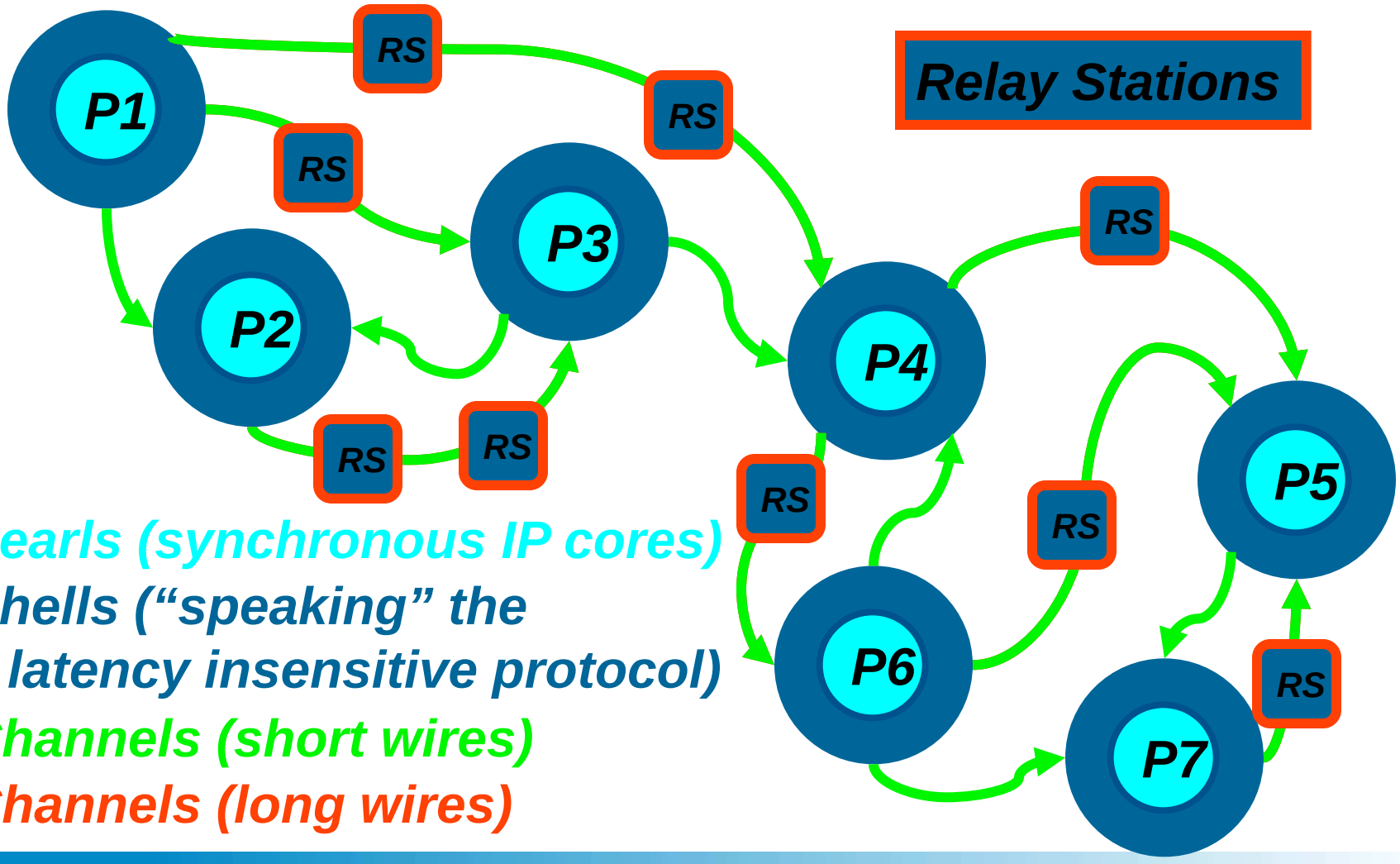
Pearls (synchronous IP cores)

Shells ("speaking" the latency insensitive protocol)

Channels (short wires)

Channels (long wires)

Latency Insensitive Design (LTI) [ICCAD99]



Pearls (synchronous IP cores)

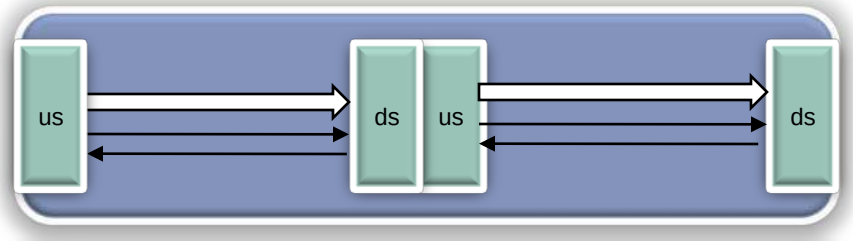
Shells ("speaking" the latency insensitive protocol)

Channels (short wires)

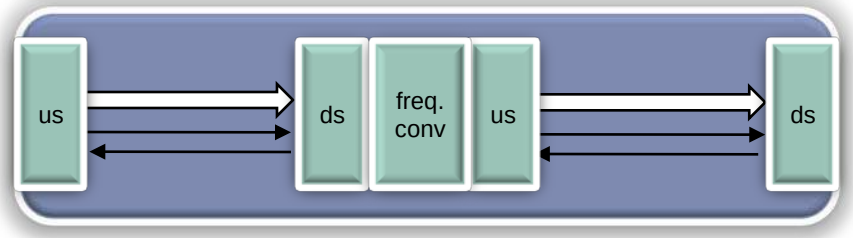
Channels (long wires)

Adaptive Link

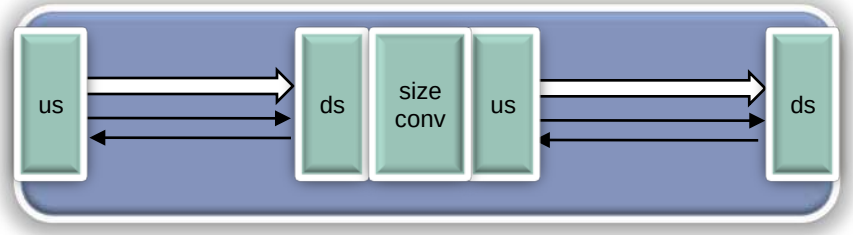
Different link configurations for different purposes



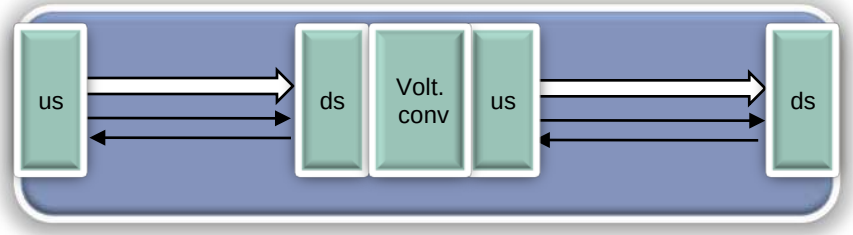
RelayStation: configuration used to break long link (backend oriented)



Alink with freq. conv: used to cross frequency domain



Alink with size conv: used to adapt flit size



Alink with volt. conv: used to support multiple voltage domain

Features and configurability

- Each VN (VN1, VN2) is individually instantiable and configurable (both DS and US are activated on the selected VN)
- Configurable flit size: flit payload size 16/18, 32/36, 64/72, 128/144 bits, plus flit extra bits
- Optional registers on credits
- IB with/without bypass capability and optional retiming stage: configurable crossing latency
- Configurable virtual channel arbitration scheme: LRU/RR, VN_priority (w/ & w/o lock packet)
- DS/US flit size conversion support (instantiates FIFOs)
- Frequency conversion support (instantiates bisynchronous FIFOs)

- When FIFOs on, can support Store&Forward:
  - Per NoC packet
  - Per NoC packet per flit threshold
- Separate FIFOs for header and payloads:
  - Optimized Header FIFO width
  - Flexibility for advanced S&F control
- Depending on traffic type, removes unnecessary Payload FIFO
- NOTE: DS/US flit sizes cannot both be lower than header width
- NOTE2: DS and US flit extra bits sizes cannot be random. Either they are zero (at least one of them) or they stay in the same ratio as DS and US flit payload sizes
- NOTE3: *four_be* conversion/management not supported for flit payload size 16/18

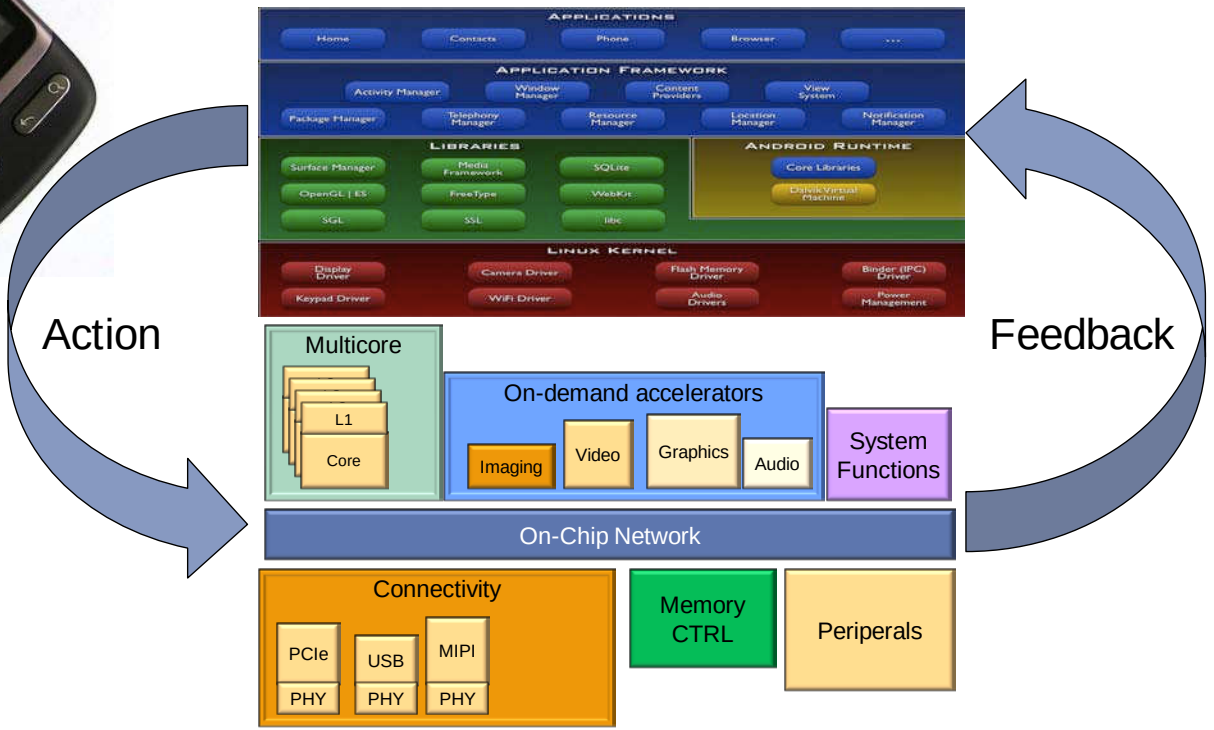
Agenda

- Setting the stage
- Overview on Spidergon STNoC
- Spidergon STNoC backbone
- User experience
- Communication Primitives & Services
- Spidergon STNoC today
- Conclusions

The User Experience

- How a product behaves and is used by people in the real world
- “every product that is used by someone has a user experience: newspapers, ketchup bottles, reclining armchairs, cardigan sweaters.” (Garrett, 2003)
- User Experience
  - the way people feel about it
  - their pleasure and satisfaction when using it,
  - looking at it,
  - holding it,
  - opening or closing it

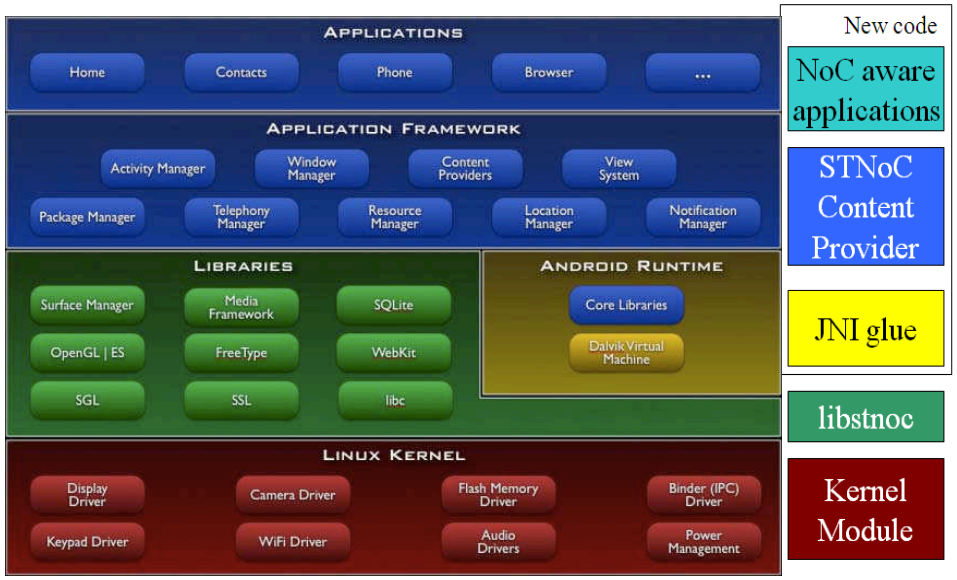
The best experience out of the box.



Agenda

- Setting the stage
- Overview on Spidergon STNoC
- Spidergon STNoC backbone
- User experience
- Communication Primitives & Services
- Spidergon STNoC today
- Conclusions

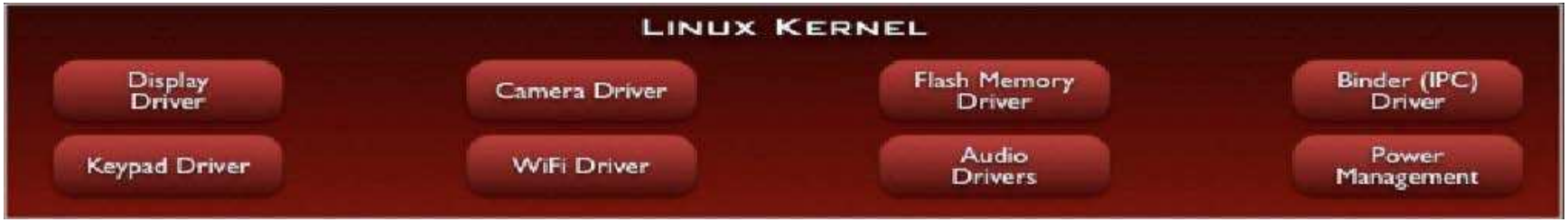
Spidergon STNoC: the Software



- A software Library (libstnoc) and set of device drivers to programme interconnection behaviour of SoC architecture and to take advantage of built-in NoC services (Power, QoS, Security, MMU, Diagnostic, etc.)

Spidergon STNoC driver

- The driver exposes in files the NoC configuration information (base address, routing type etc)
- Routing, QoS, and security information
  - Through writable files, writing to the files updates the configuration registers
- Last Register status
- Possible destinations
- Alternative paths for source routing



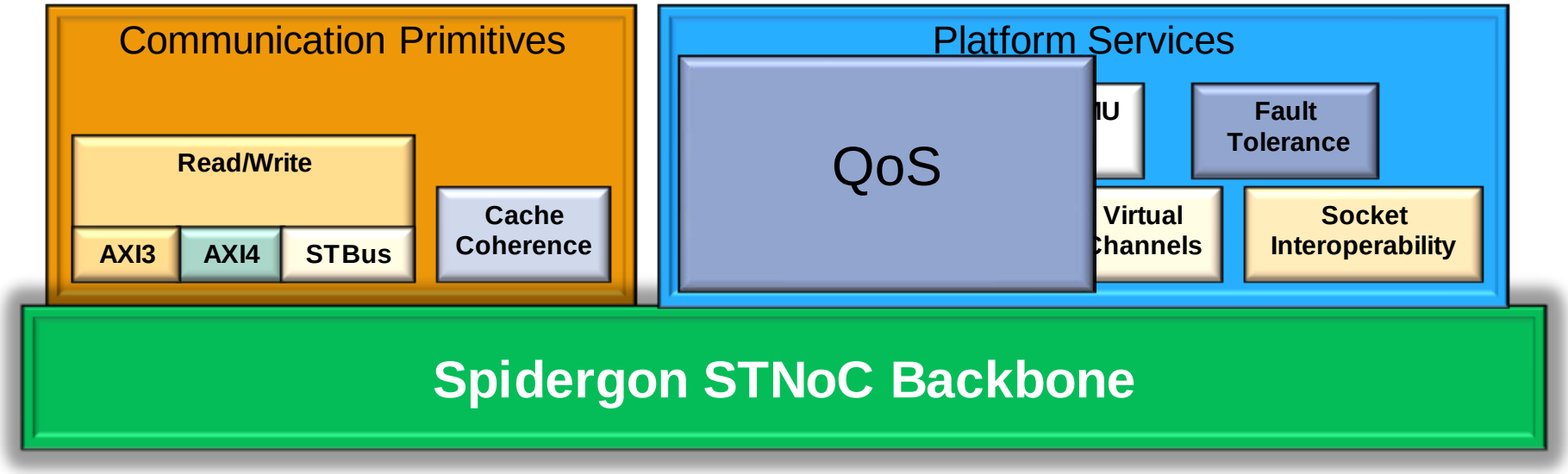
Spidergon STNoC Library: Libstnoc

- Libstnoc provides a high level C API, abstracted from the details of register format
- A simple interface to allow applications to optimize the NoC at any time for their requirements.
- High level get and set functions for routing, QoS.
- Error handling/valid data checks before applying to the registers.
- Higher level functionality: profile switching / requirements based reprogramming of the NoC.

NoC aware applications on Android

- The Conent Provider, provides access to STNoC functionality to the whole Java based Android Application Framework.
- Possible Applications.
  - GUI on-device debug interface for the NoC.
  - Parameters adjustment when receiving a phone call, in flight mode, in standby etc.
  - Adjustments for performance or low power.

Platform Services: An example

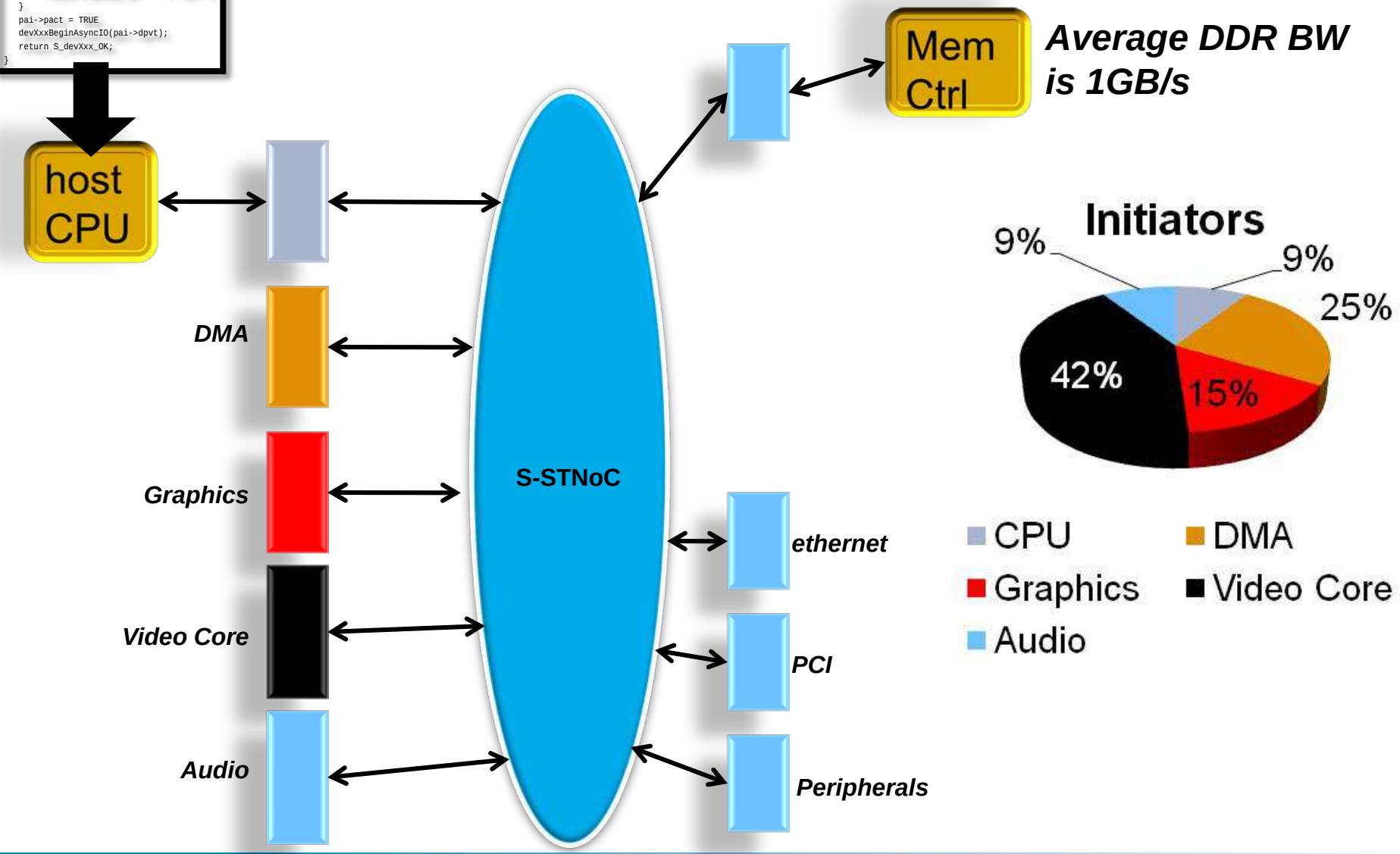


Dynamic resource usages: the web browser

Dynamic resource usages: face time

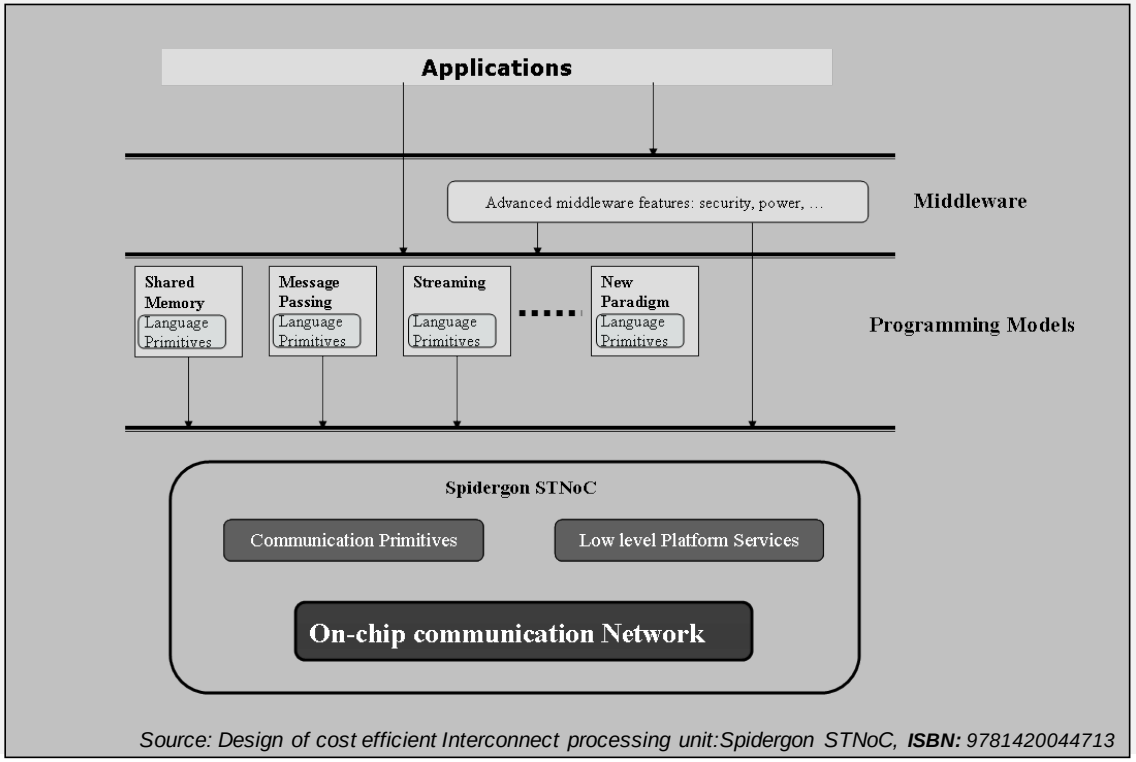
Service: Programmable QoS

```
long devXxxRead (struct aiRecord *pai) {
   if (pai->pact) [
        return S_devXxx_OK;     /* zero */
   }
   pai->pact = TRUE
   devXxxBeginAsyncIO(pai->dpvt);
   return S_devXxx_OK;
}
```



Towards IPU

- IPU = Interconnect Processing Unit
- IPU Services are implemented in hardware and/or software
  - Communication Primitives
  - Low level Platform Services



Source: Design of cost efficient Interconnect processing unit:Spidergon STNoC, ***ISBN:*** *9781420044713*

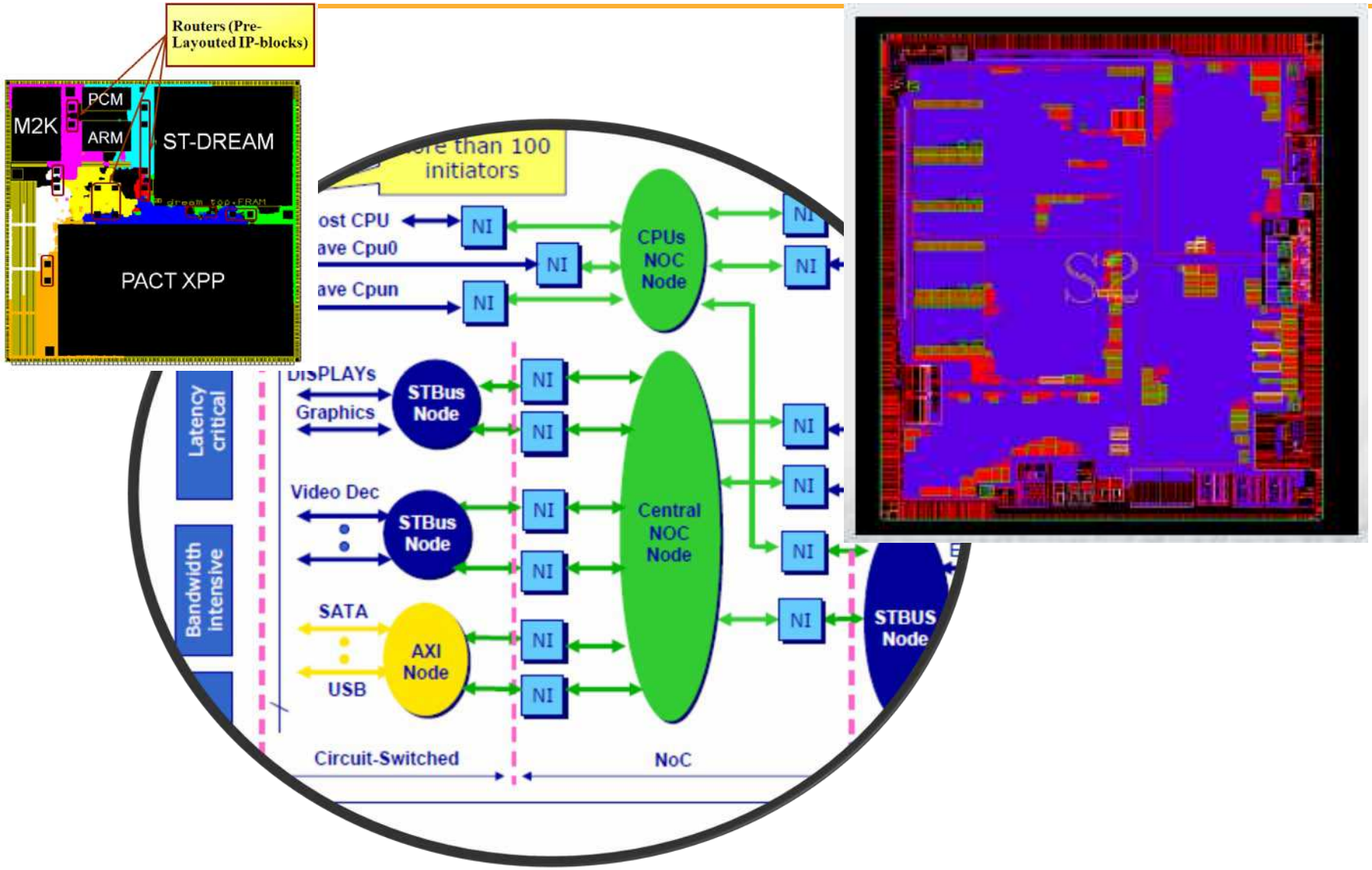
Agenda

- Setting the stage
- Overview on Spidergon STNoC
- Spidergon STNoC backbone
- User experience
- Communication Primitives & Services
- Spidergon STNoC today
- Conclusions

Where is Spidergon STNoC today?

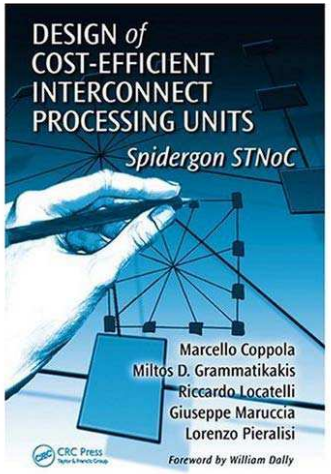
- Spidergon STNoC is not only a standard NoC technology but it is an innovative technology able to provide real product differentiation through a set of smart hardware/software software services
- ST on-chip communication network strategy in following years in consumer, multimedia, mobile etc.
- External company licensing (eg. STE)

Spidergon in real SoC: The hybrid NoC



For more info

- Book published by CRC press
  - Spidergon STNOC book
  - **ISBN:** 9781420044713
  - **Publication Date:** September 2008



- Web based info from European/Catrene funded projects
  - 3DIM3
  - COMCAS
  - MODERN
  - SHAPES
  - …..