Co-Synthesis of a Configurable SoC Platform based on a Network on Chip Architecture

Mário P. Véstias & Horácio C. Neto
INESC-ID (PROSYS), Lisbon, Portugal

ASP-DAC'06
Yokohama, Japan
January 24-27, 2006

Outline

1. Introduction and Motivation
2. What Exists?
3. Our project proposal
4. Target architecture
5. Design methodology
6. Application example
7. Conclusions
8. Future work

1. Introduction and Motivation

- Development of systems for high-capacity data processing on configurable platforms
- Working on architectures and design methodologies for SoC design
- Developed a platform-based methodology with IP and architecture reuse.

Problems with the interconnection architecture (bus-based) :

- Does not support the efficient interconnection of many IP in a structured way.
- Increasing connection difficulties – lack of scalability
- Need reuse of interconnection!

Solution => Network on Chip

2. What exists

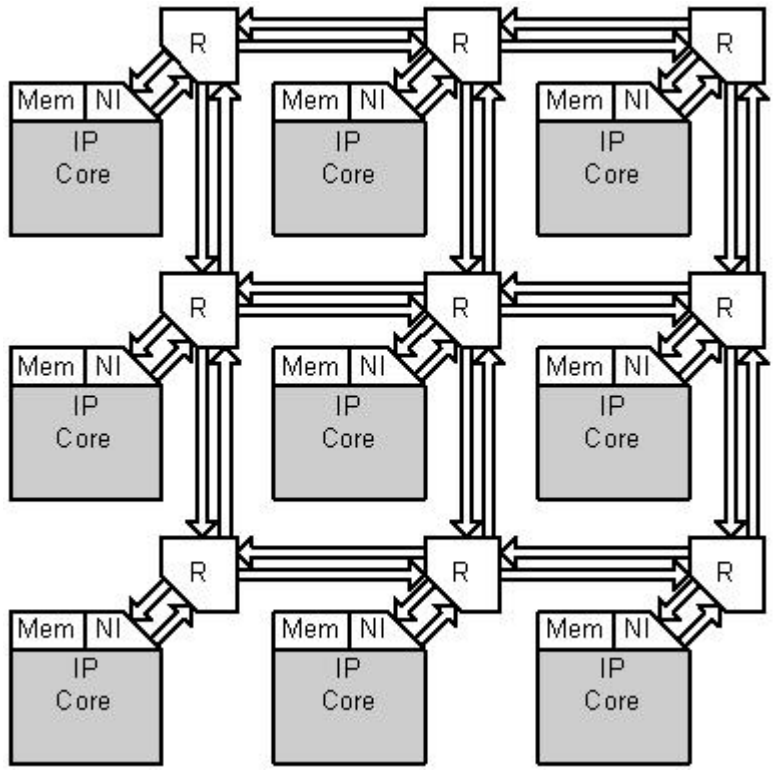
- There exists some work on NoC development
  - Topologies, structures, protocols, etc.
- There exists lots of work on cosynthesis and codesign, but for bus-based architectures
  - Allocation, mapping, scheduling for Hw/Sw systems
- It is time to adapt many of these techniques and ideas to develop SoC systems based on the NoC paradigm
  - Some tools already exist, but not a complete co-synthesis framework

3. Our project proposal

- In a first step, our project consisted on:
  - Develop a co-synthesis methodology based on a NoC architecture
  - Study the impact on design productivity
  - Study the impact on design complexity
  - Study the impact on design quality

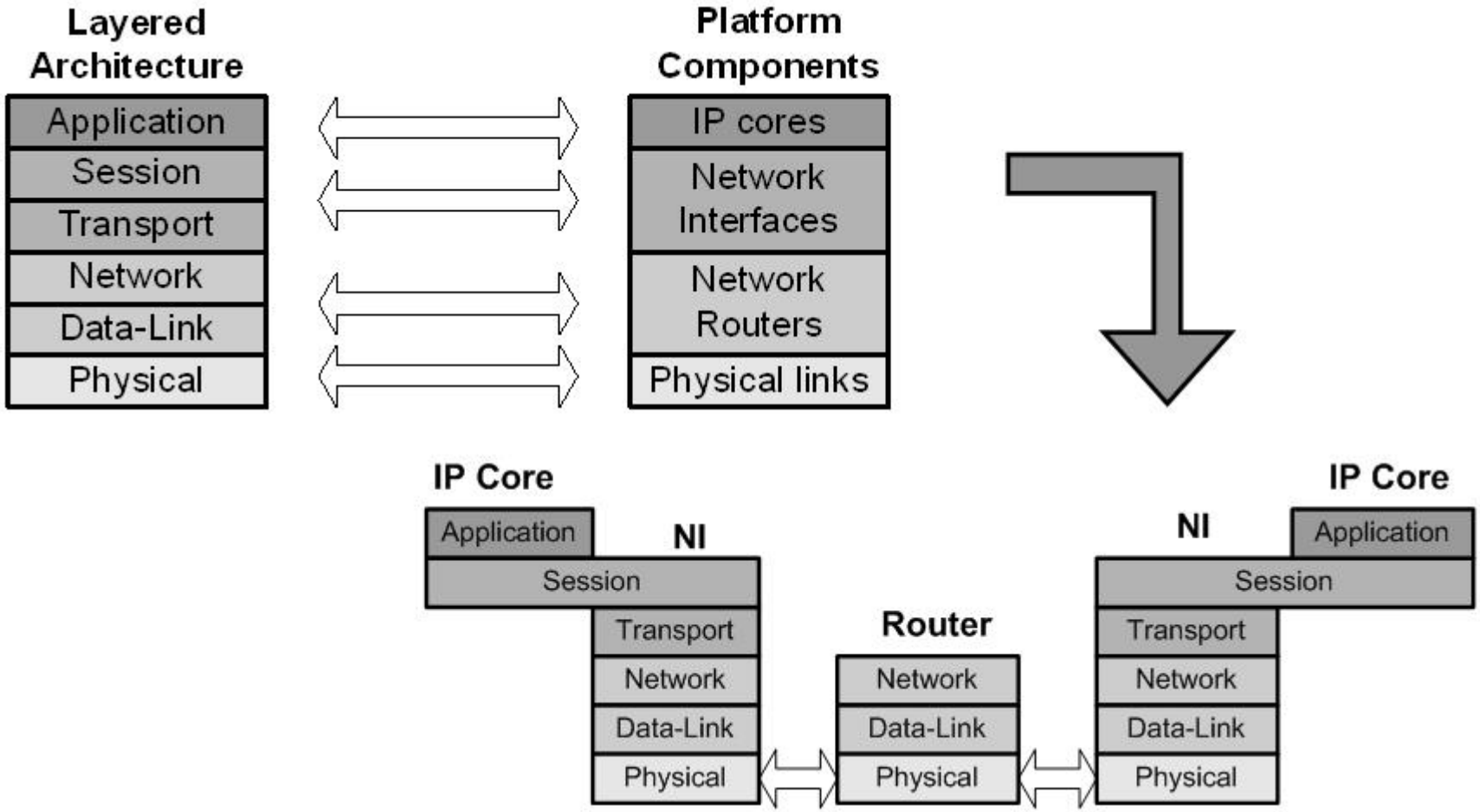
4. Target architecture

- NoC:
  - 2D-mesh topology
  - Routing, flow-control, switching, arbitration, buffering
- Parameterization:
  - 2D-size
  - Type of core
  - Link size



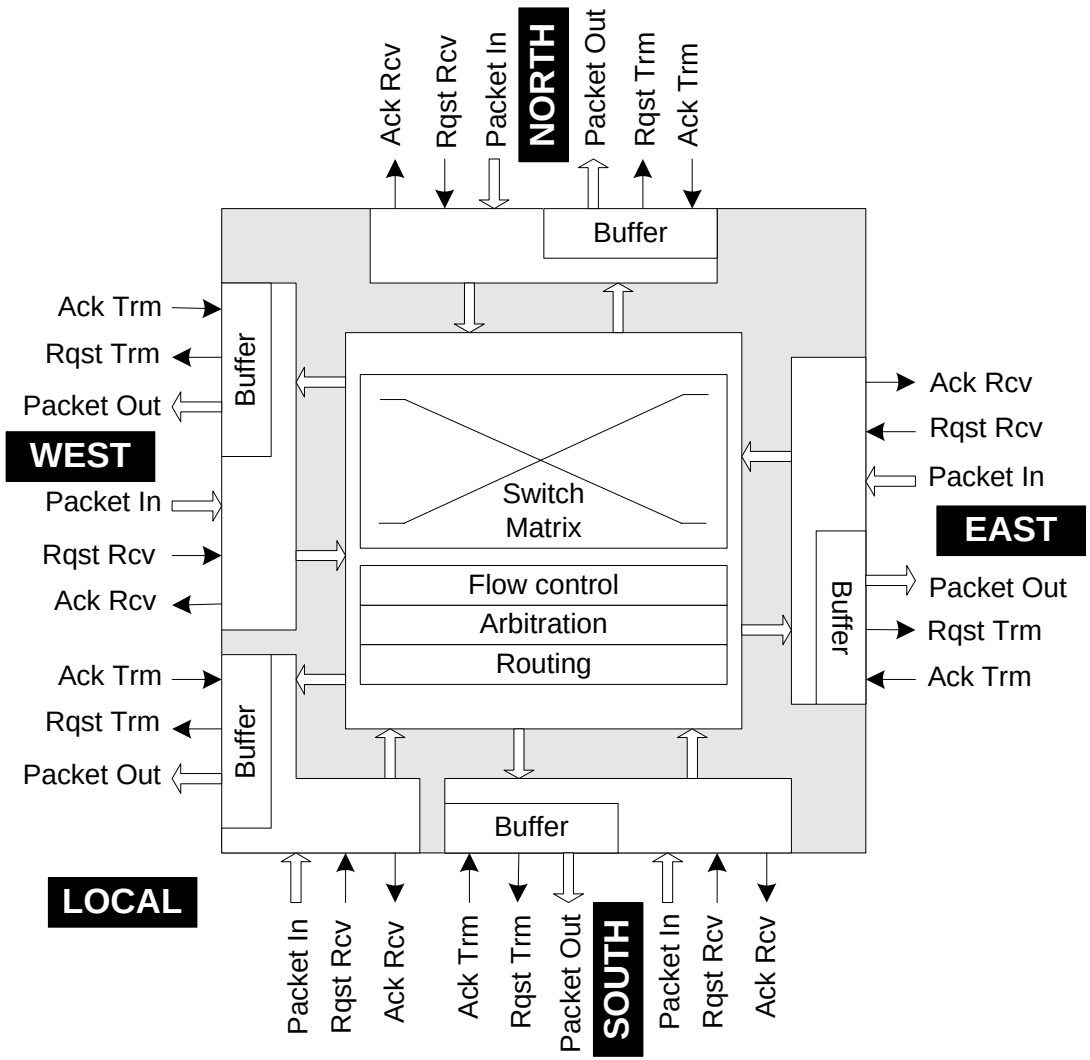
4.1 Target architecture - behavior

NoC behavior similar to the OSI communication architecture



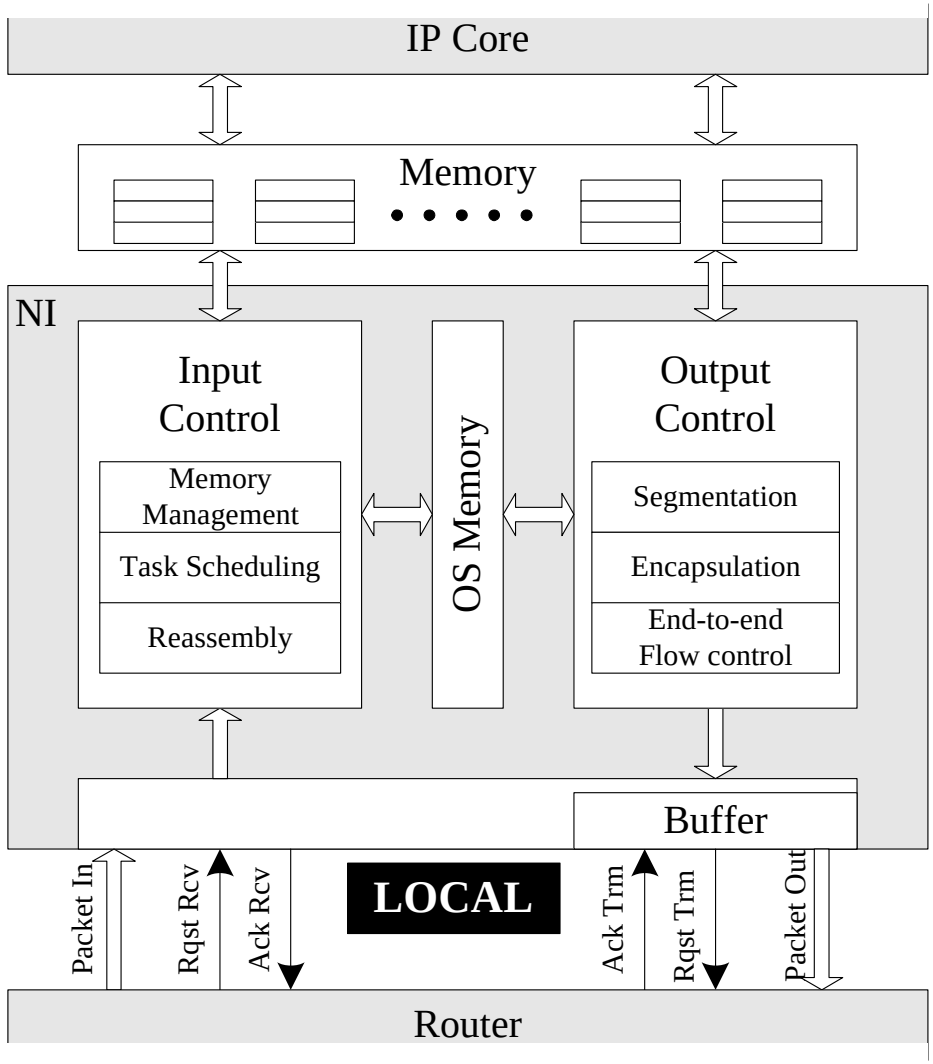
4.2 Target Architecture - Router

- Routing – XY algorithm
- Flow-control: 2 way handshake
- Switching: store and forward
- Arbitration: round-robin
- Buffering: output



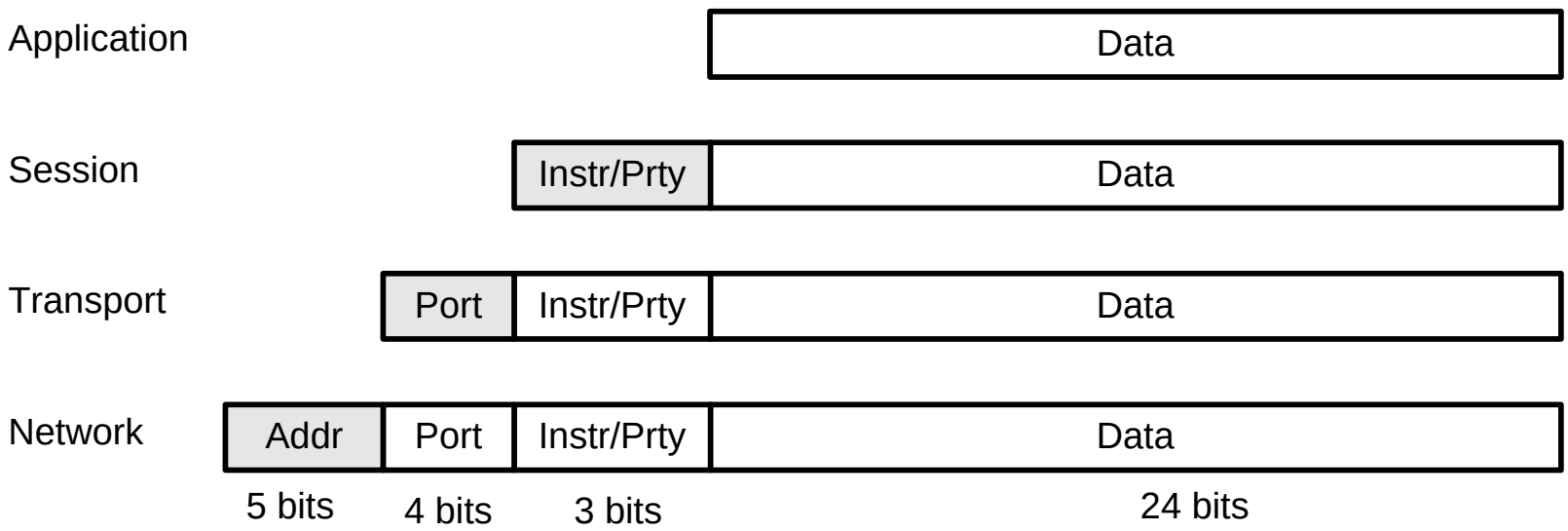
4.3 Target Architecture - Network interface

- Shared Memory comm
- Functions:
  - Memory management
  - Task scheduling
  - Reassembly
  - Segmentation
  - Encapsulation
  - Flow-control



4.4 Target Architecture - Packet structure

- Data packets
- Configuration packets
- Token packets



4.5 NoC Characterization - performance

- **Link latency** (LL) – 1 cycle;
- **Resource generation latency** (RGL) – 4 cycles;
- **Resource reception latency** (RCL) – 5 cycles;
- **Resource to resource latency** (R2RL) - …;
- **Resource to resource bandwidth** (R2RB) – f/5 packets/s.

Edge delay = $\frac{RGL + RCL + LL \times NR}{f\,(\text{working frequency})}$

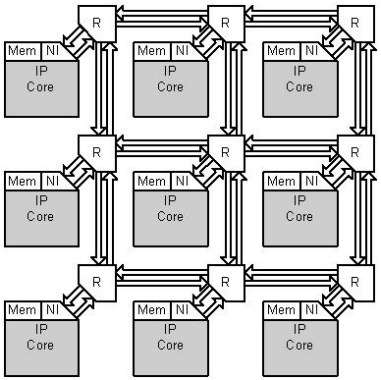
Edge delay$_{\text{pipe}}$ = $\frac{1}{R2RB} \times (NR + NP)$

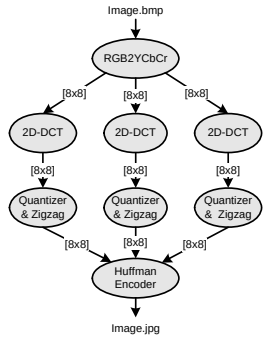
Can forward up to 750 Mpackets/s => 6Gbps

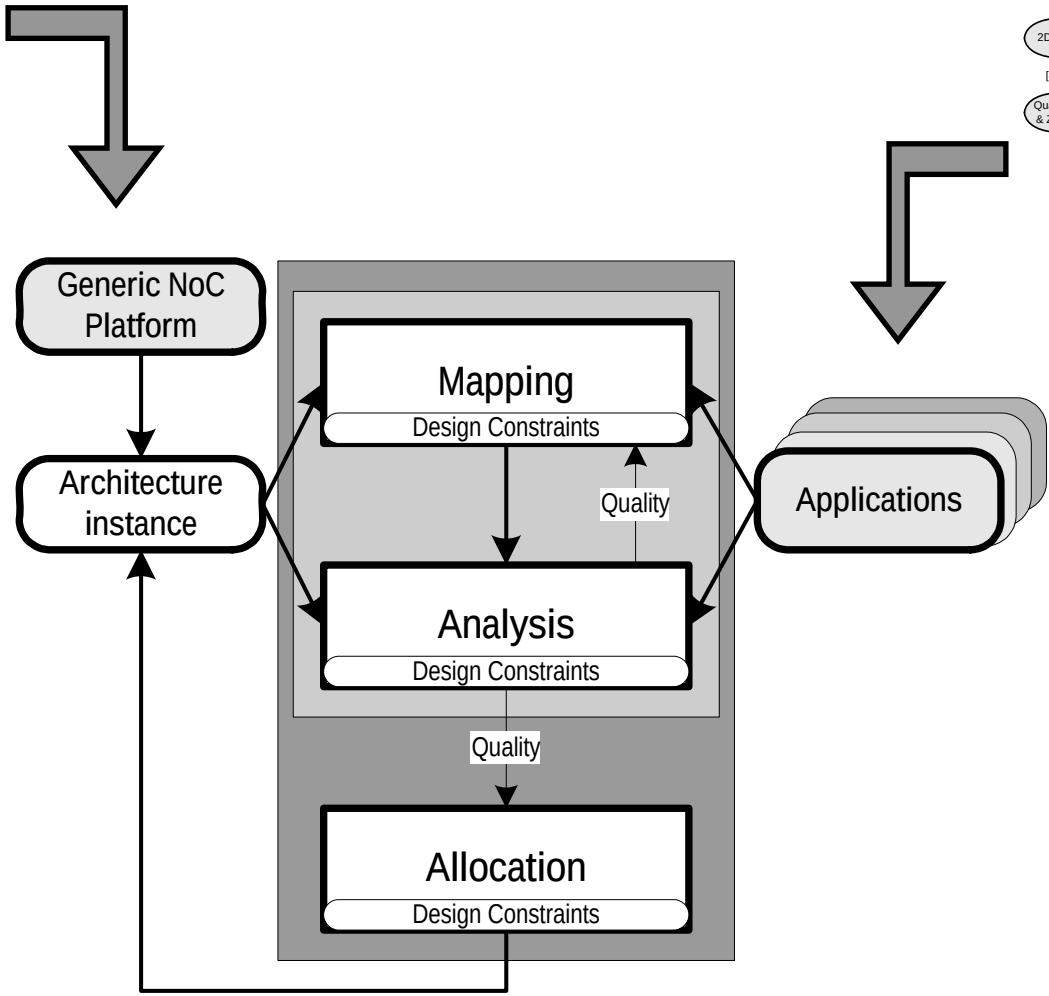
4.6 NoC Characterization - area

Block	Size (slices)	BRAM	% XC2V6000
Router (8bits)	189	0	0,56
NI	121	1	0,36

5. Design Methodology



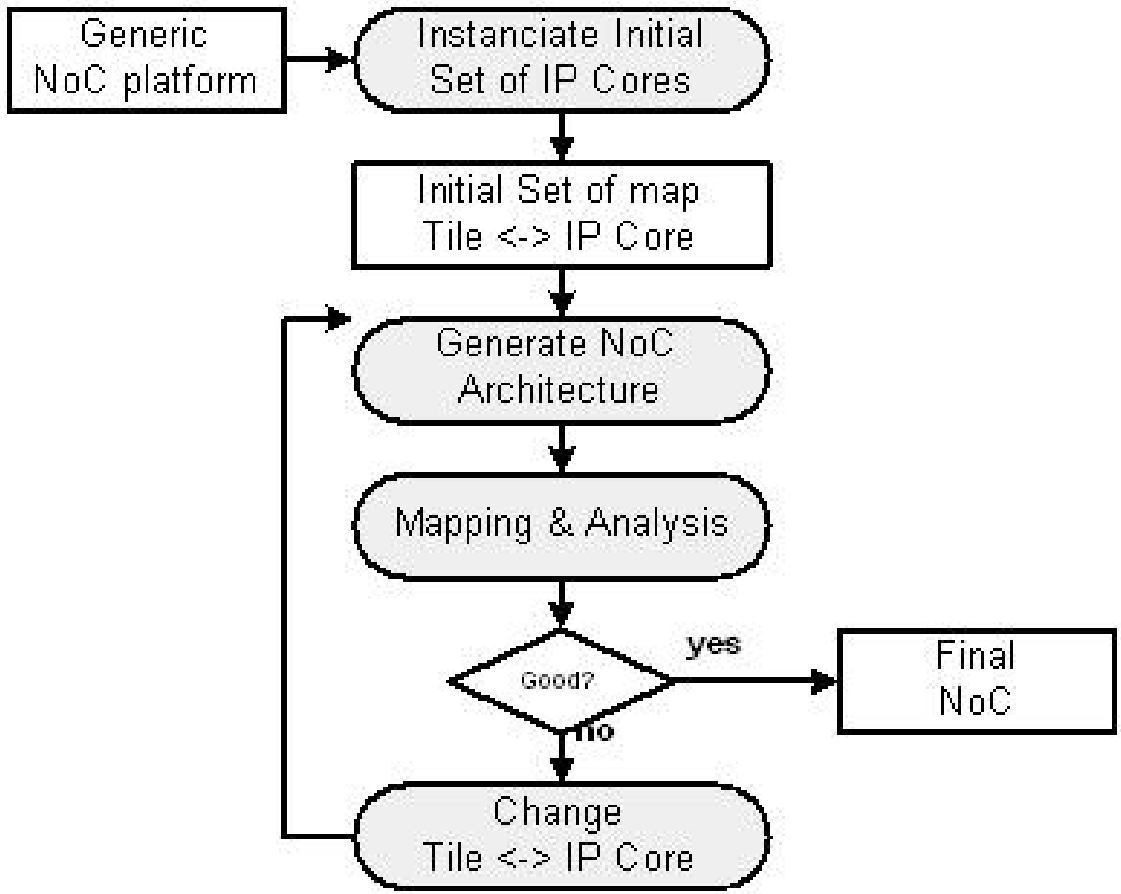




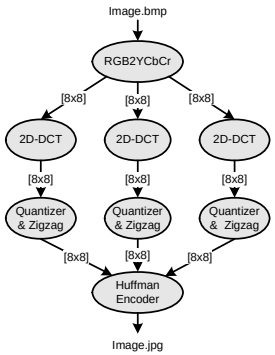
5.1 Design Methodology - Allocation

- **Size of topology**
- **Type of cores**

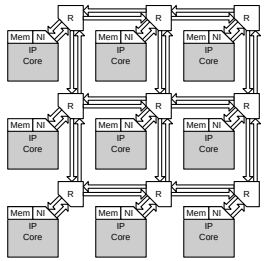
SA Algorithm

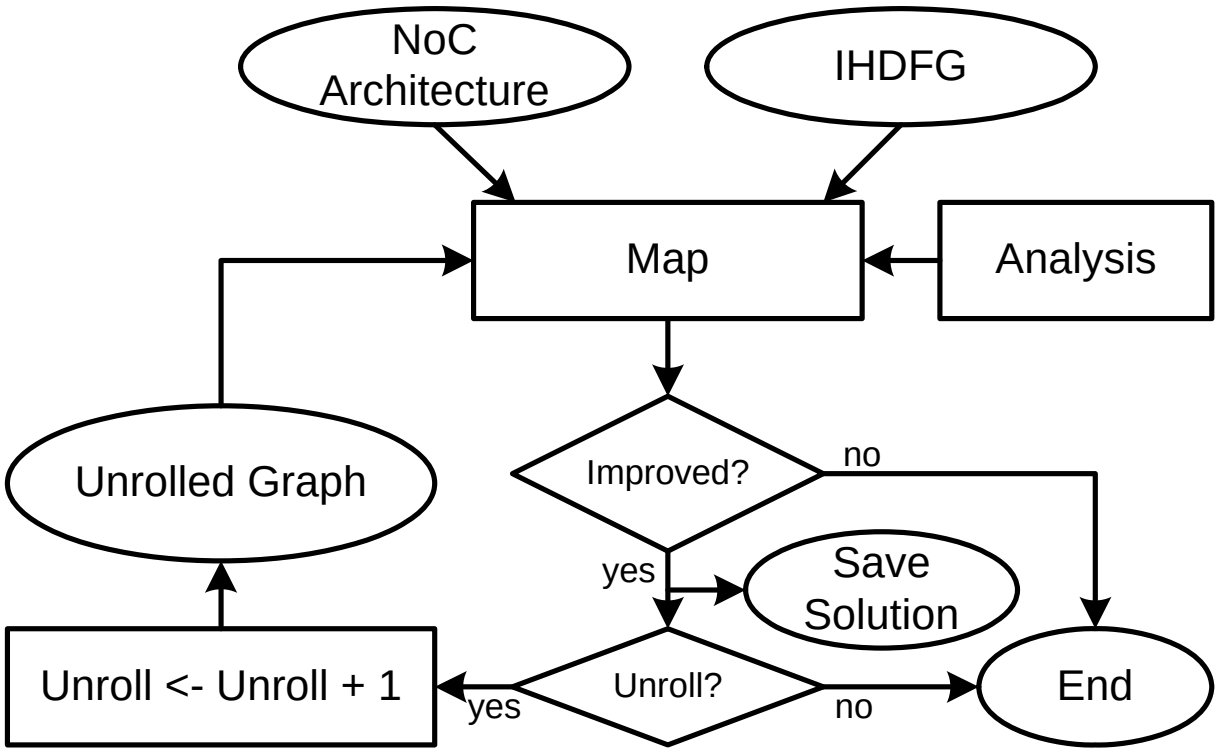


5.2 Design Methodology - Mapping

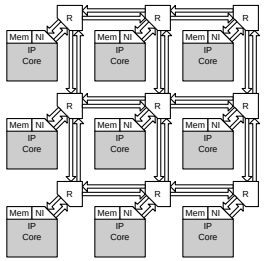


Assigns task, data transfer and variable to core, link and memory - SA Algorithm





5.3 Design Methodology - Analysis

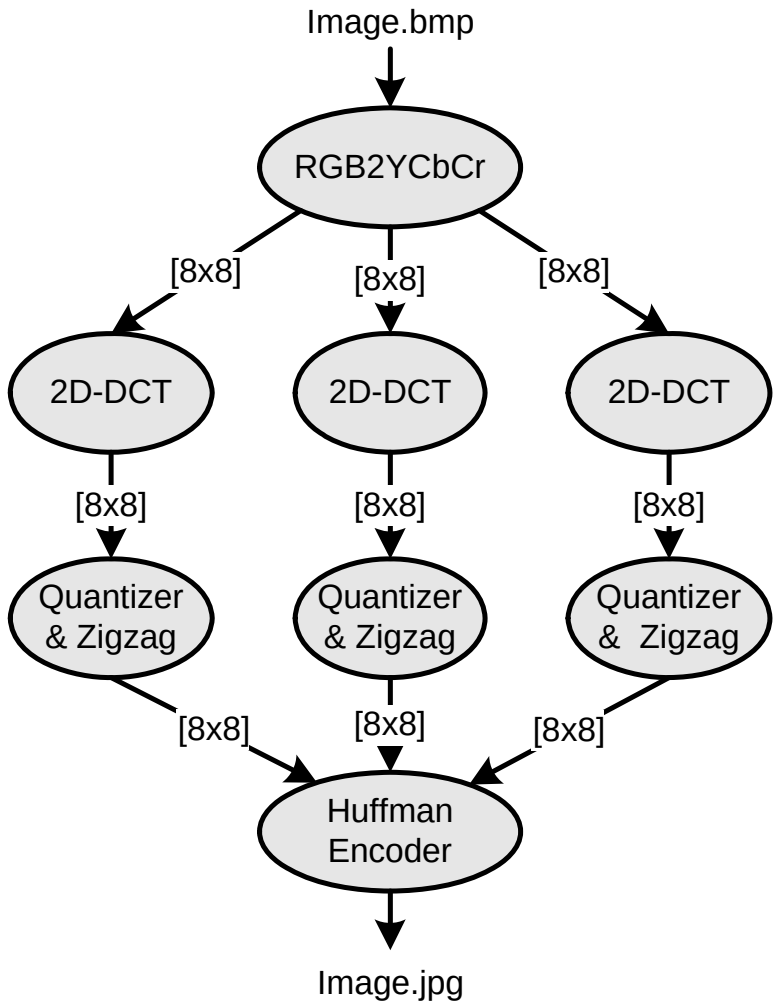


Quality = $C_{throughput}$ + $C_{throughput}$

$$Cx = \begin{cases} K \times \dfrac{Ci(P)}{\overline{Ci}}, & \text{without constrain} \\ Ka \times \max\left(0, \dfrac{(Ci(P) - Ci)}{Ci}\right), & \text{with constrain} \end{cases}$$

6 Application Example - JPEG

JPEG algorithm for color images with block size 8 x 8

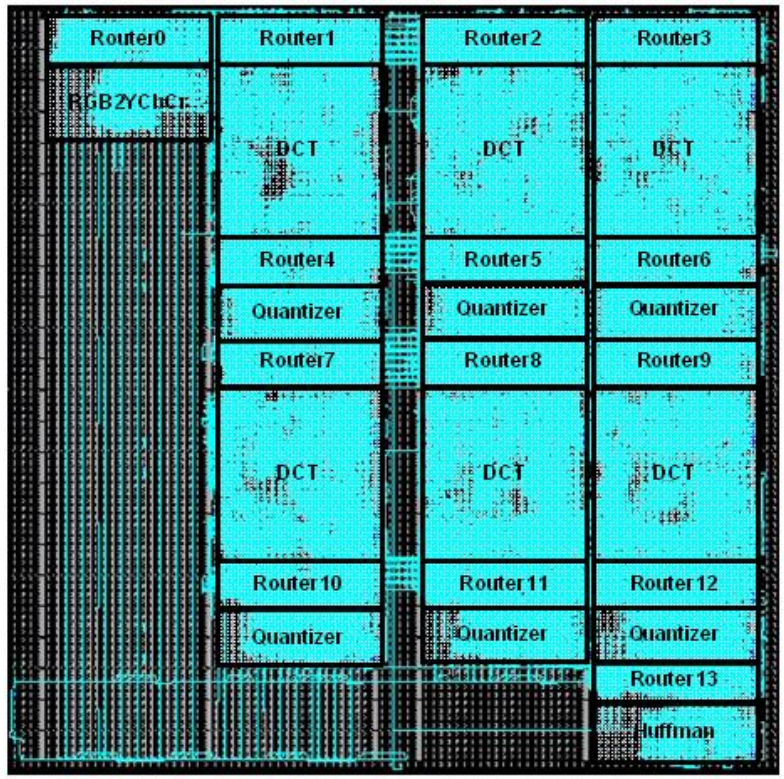


Task	Size (slices)	BRAM	Latency
RGB2YCbCr	204	0	64
2D-DCT	1612	1	168
Quantizer	312	1	64
Huffman	176	1	192

6.1 Application Example - Results

The solution processes two blocks of [8×8]×24 bits in 3.8 µs => 800 Mbps (VirtexII XC2V6000)

Image size	HW NoC sec/fps	Pentium 4 at 1.7 GHz	HW bus sec/fps
640×480	0.009 (108)	0.046	0,055
800×600	0.015 (67)	0.071	0,086
1024×768	0.024 (42)	0.110	0,14



7 Conclusions

- Design complexity – easier to design due to scalability, efficient interconnection of IP and communication/performance “independence”
- Design productivity – design methodology accomplishes goals more efficiently
- Design quality – better for communication intensive applications. Better throughput; latency(?). Hybrid structure with routers and buses must be considered

7 Conclusions

- A research over a set of IP cores concluded that smaller routers must be implemented to improve cost, power consumption and performance;
- Many NoC parameters must be part of the design methodology to improve final system quality – buffer size, switching capacity, arbitration policy, topology, etc.

8 Future work

- Considering a more generic topology with routers consisting of more than 1 local port and sharing of router resources to improve cost, performance and power dissipation;
- Increase the set of configurable parameters of the NoC architecture in the design methodology;

Co-Synthesis of a Configurable SoC Platform based on a Network on Chip Architecture

Mário P. Véstias & Horácio C. Neto
INESC-ID (PROSYS), Lisbon, Portugal

ASP-DAC'06
Yokohama, Japan
January 24-27, 2006