Automated derivation of NoC Communication Specifications from Application Constraints

Samuel Evain, Jean-Philippe Diguet, Milad Elkhodary, Dominique Houzet

Automated derivation of NoC Communication Specifications from Application Constraints

Samuel Evain, Jean-Philippe Diguet and Milad El Khodary
Université de Bretagne Sud,
LESTER, FRE 2734 CNRS,
Rue de Saint-Maudé,
56325 Lorient Cedex - France
Jean-Philippe.Diguet@univ-ubs.fr

Dominique Houzet
Université de Rennes 1,
IETR-INSA, UMR 6164 CNRS,
20, avenue des Buttes de Coësmes,
35043 Rennes Cedex - France
Dominique.Houzet@insa-rennes.fr

***Abstract*— This paper focuses on the highest step of our NoC design flow, which addresses the efficient deployment of real applications over an ad hoc NoC. At this level we propose a methodology and a tool to decide the NoC parameters and to generate the path coding within network interfaces for guarantied and best effort communications. The originality of our approach is based on two points. First our tool includes a derivation technique to obtain NoC communication constraints (latency, bandwidth) from application designer knowledge (application throughput). Secondly, the decision tool explores a 3D graph (t,x,y) for path allocation while taking into account mutual exclusion and global latency for FIFO minimisation under time constraints. This paper illustrates the first point. Two real applications: smart camera and multiprocessor turbo-decoder are presented to illustrate the design flow.**

I. Introduction

A lot of efforts have been performed in the domain of Network-on-Chip (NoC) design in the last seven years. The first kind of work was dealing with proof of concepts, topologies and workload simulations. Then researchers have explored different topics such as virtual channels for guarantied traffics, network interfaces, IP mapping, asynchronous communications, adaptivity, security and some current new directions, like 3D design for heterogeneous huge system on chips, bring out some new opportunities. Few CAD tools are proposed for NoC design decisions, the first category deals with a library of NoC components and simulations tools to verify data-rate constraints [1]. The second category aims to guaranty real-time constraints with time division multiplexing (TDM) NoC accesses while considering individual constraints (latency, bandwidth) for each communication individually ([2], step 4 of our flow). Today some additional efforts are required to adapt CAD tools to the real needs of system designers. The NoC design space is in practice intractable if all problem dimensions are considered simultaneously, these dimensions would at least include : topology, mapping, media access policy (best effort (BE) traffic, Guaranteed Throughput (GT) traffic)), TDM scheduling for GT traffic, path allocation, latency and bandwidth constraint checking. From a designer point of view, the problem never starts from scratch but from a set of heterogeneous communicating IPs (dedicated hardware, memories, processors) and application throughput constraints. First, it appears that the topology optimization is not currently a real issue since few alternatives have to be tested in reality. Secondly, the mapping exploration is quite constrained since IP usually have heterogeneous sizes and limited real mobility within the SOC. We believe, according to designer needs and knowledge, that an interactive tool is required for mapping, topology and communication policy selections (1). A CAD tool must focus on real tedious and error prone design steps. To our point view, these steps are (2) the derivation of individual communication constraints from the application specification, (3) the TDM table design, (4) the slot / path decision including buffer sizing and, according to all previous features, (5) NoC code generation. Our complete design flow described in figure 1 is based on this approach. The paper mainly focuses on step 2 of our flow. In section II, we present the state of the art related to it. In section III we briefly describe our complete NoC design flow. In section IV we deeply detail steps 2. In section V, we apply our flow to two realistic applications, a smart camera and a multiprocessor turbo decoder. Finally we conclude.

II. State of the Art

In [2] is proposed a design flow for GT implementation in NoC. It is based on iterative phases of NoC generation, NoC configuration and NoC performance verification. It uses TDM slots for GT. Designer must specify required bandwidth, burst size, latency constraints and GT/BE traffic class for each individual read/write transaction. This assumes a deep study of the application communications by the designer to provide those informations. In [3], timing analysis of the application is used to prove guaranty of traffic. Authors consider the utilization of communication dependence and computation graphs extract communications that may or not be in conflict in order decide a relevant mapping. It is based on a simple XY routing technique, so the path is unique for each IP mapping, which is computed with a simulated annealing algorithm. This approach only focuses on mapping. Our approach deals with application specification, to derive communication constraints and to identify mutual exclusions to guide path allocations, so it is based on application analysis to find a proper solution.

III. Complete NoC Design Flow Overview

Figure 1 gives an overview of the NoC design Flow. Our NoC model enables the implementation of two kinds of communications: best effort (BE) and guarantied traffic based on a TDM technique. The first step is an interactive IHM that enables the designer to rapidly specify the application and the NoC parameters. The application is specified with a set of characterized communication tasks, with application throughputs and if necessary with links between mutual exclusive communications. The NoC knobs are related to the NoC topology which can be ad hoc, the router parameters (ports, routing policy, arbiter, Ě), the Network Interface (NI) and wrapper specification (slave, master, bus standard) and the IP mapping namely the IP/NI association. If real-time constraints are required, then related communications are implemented with virtual channel (guarantied traffic). The second step is automated and deals with derivation of local latency and bandwidth constraints for each unidirectional communication from application I/O throughputs. The important issue, which is always omitted in NoC design flows, is in practice necessary for applying steps 3 and 4. This work is not trivial since firstly different local decisions are possible two meet global constraints and secondly latency, bandwidth and TDM table size are strongly dependent. Moreover read operations imply two types of heterogeneous communications: the read command and the data response for with two distinct set of constraints must be define. Thus, the second step first transforms communication tasks into unidirectional ones. This aspect is required for read operations that need a lightweight forward communication for sending a read command and a backward communication for receiving data. Then we produce, for each communication with a guarantied traffic, the minimum bandwidth and a set of rules for latency/bandwidth checking. The third step computes the minimum TDM table size required for implementing GT communications and a minimum bandwidth for all BE communications. It provides minimum latencies for each communication to the next step. The fourth step explores the time (TDM slots)-space (NoC paths) space in order to allocate time slots to each GT communications. It provides the next steps with a complete NoC specification. Due to space restriction, this step is not detailed in this paper. The fifth and last step is the VHDL code generator, some additional C API codes are also provided for interfacing the NoC component with IP compliant with the OPB bus standard bus and/or Xilinx FSL ports.

IV. Application Specification Formalisation

On the one hand, in telecom and image processing domains, an application is usually specified as a set of communicating tasks with global input/output time constraints. On the other hand Latency and Bandwidth constraints are necessary for mapping guarantied traffic communications within a NoC. So, local constraints have to be derived from global ones. However these issues are not trivial since design decision are strongly dependent. Thus, for solving the TDM sizing problem, latency

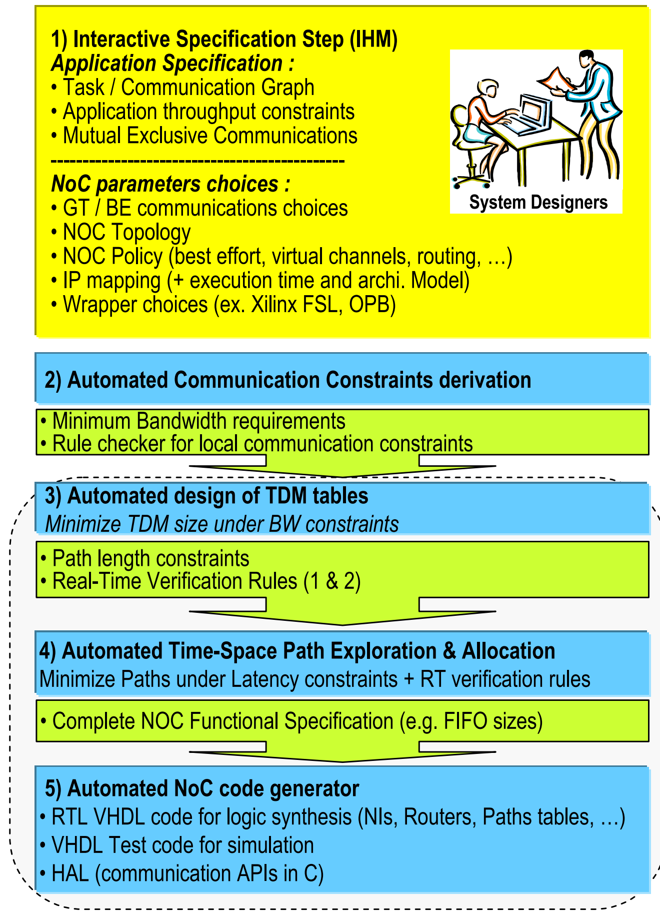

Fig. 1. NoC Design Flow

and bandwidth constraints are required for each communication, however latency depends on TDM specifications and on bandwidth, which is also related to the latency. We face a usual CAD problem of decision ordering. Our approach is based on starting assumptions that aim to minimize the most critical NoC parameter namely the FIFO global size.

A. Real-time verification rules

In telecom and multimedia domains, communication dominated applications can be specified as a graph of tasks exchanging data through a NoC. In Fig.2 a simplified view is given, a chain of four tasks (1-4) must be executed within a period T. In practice, an application can be specified as a set of chains tasks running within specific periods. Without loss of generality, we illustrate constraint computation with a single chain of tasks. Two constraints have to be considered: the initialization constraint and the cadence constraint.
The initialization constraint addresses the delay for reading the minimum amount of data a task Ti needs before starting computation; This initialization is computed while adding the initialization delays from the input to the output. A task Ti requires three kinds of communication as illustrated in Fig.3. CFRi is the communication related to the read instruction, CBRi is the communication for reading data consumed by Ti,

CFWi is the communication for writing data produced by Ti.

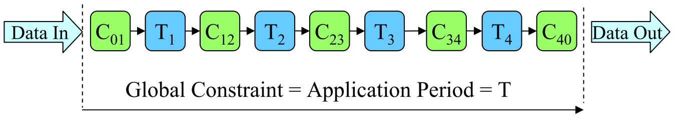

Fig. 2. Communication and task chain

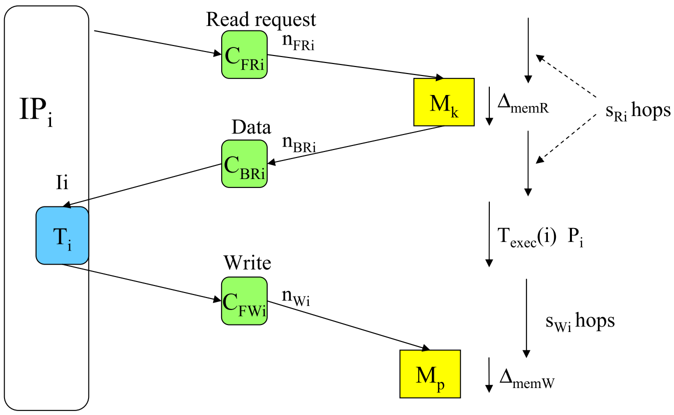

Fig. 3. A task and its communications

- T: application period;
- nFR_i : number of words for CFR_i;
- nBR_i : number of words for CBR_i;
- nFW_i : number of words for CFW_i;
- P_i : number of T_i iterations during T;
- I_i : number of read communications before starting T_i;
- $\Delta memR$, $\Delta memW$: memory access time for read and write operations respectively;
- SR_i : number of hops for a read operation;
- SW_i : number of hops for a write operation;
- $Texe(i)$: task execution time for one iteration;
- O_i : number of task iterations to provide the required number of words to initialise the subsequent tasks.

For all tasks j consuming data produced by task i :

$$O_i = \left\lceil \frac{I_j nBR_j}{nFW_i} \right\rceil \tag{1}$$

Thus we can define an initialization delay for a task Ti to start as Tinit(i) such as :

$$\begin{aligned}
&T_{init}(i) = \\
&Max_{p \in Prev(i)}(T_{init}(p)) \\
&+F_T(nFR_i, K_{FRi}, SR_i) \\
&+\Delta_{memR} \\
&+F_T(I_i nBR_i, K_{BRi}, SR_i) \\
&+T_{exe} \\
&+\max\{Texe(i) - F_T(nFW_i, K_{FWi}, SW_i); 0\}(O_i - 1) \\
&+F_T(O_i nFW_i, K_{FWi}, SW_i) \\
&+\Delta memW(i)
\end{aligned} \tag{2}$$

Where P(i) is the set of T_i predecessors and F_T the communication delay in the NoC. F_T depends on the TDM table in the Network Interface (see Fig.4), the path length and the number of data to be transmitted, it is defined as follows :

$$F_T(n, k, p) = \lceil \frac{n}{K} \rceil (N - K) + n + pL_S \tag{3}$$

where :

- L_s is the number of words within a slot;
- N is the number of words in TDM table ($N = L_s|S|$);
- $|S|$ is the number of time slots in TDM table;
- K is the number of slot reserved for the considered communication;
- p is the path length in hop counts.

This model is generic and can be applied to different communication models such as shared memories or direct communications between IP FIFOs, in that last case nFR_i is null.

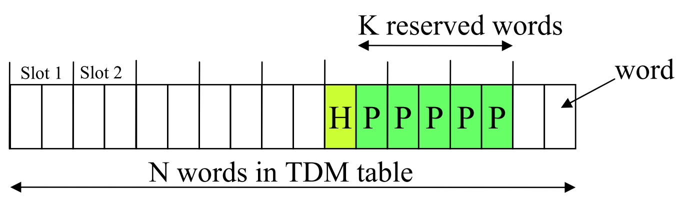

Fig. 4. TDM

The first verification rule is given in Eq.4 and is related to the initialization constraints. These rule is necessary but not sufficient to guaranty real-time constraints since cadences must be verified to ensure that data can be produced on time after each iteration.

Rule 1 :

$$T_{init}(j) + F_T(nFW_j, (P_j - 1)nFW_j, SW_j) < T \tag{4}$$

The cadence verification is related to the ability of the system to consume and produce data according to application cadences. Basically it means that the bandwidth allowing to each task to proceed must be compliant with its communication requirements. We assume that, in a steady data flow, the latency is hidden since a read (resp. write) communication requests can be launched before the previous ones have proceeded. Different cases must be considered depending on the IP architecture models where read communications, write communications and task execution can be executed sequentially or simultaneously. For simplicity sake we present a single rule compliant with a full sequential model. In that case we obtain :

$$\begin{aligned}
T_{read}(i) &= \Delta_{memR} + F_T(nBR_i, KBR_i, SR_i) \\
T_{write}(i) &= \Delta_{memw} + F_T(nFW_i, K_{FWi}, SW_i)
\end{aligned} \tag{6}$$

Rule 2 :

$$T_{read}(i) + T_{write}(i) + T_{exec}(i) < T/Pi \quad (7)$$

B. Bandwidth and latency verification rules for TDM sizing

The TDM table must be defined before guarantied traffic path allocations. The TDM is built in order to meet latency (Li in cycles) and bandwidth (BWi in words per cycle) constraints. Without any information about the TDM table, the constraint for a communication Ci can be specified as (if we assume having a sequential architectural model namely the worst case):

$$L_i + \frac{P_i n_i - 1}{BW_i} < T - \delta_m P_i T_{exe}(i) \quad (8)$$

$$\Leftrightarrow \qquad BW_i > \frac{P_i n_i - 1}{T - \delta_m P_i T_{exe}(i) - L_i}$$

$$BW_i^{min} \simeq \frac{P_i n_i - 1}{T - \delta_m P_i T_{exe}(i) - S_i} \quad (9)$$

where n_i is the number of words transmitted during each C_i iteration, δ_m equals 1 (resp.0) if computations and communications are performed sequentially (resp. simultaneously), S_i is the shortest path. Guarantied traffic for telecommunication and image processing are usually dominated by communications and deal with periodic communications $(P_i \gg 1)$ of large amount of data ($n_i \gg 1$). Moreover, the TDM access delay is masked by decoupling buffers. It means that L_i could be neglected compared to the bandwidth term, however to be safe we set $L_i = S_i$, which is the shortest path delay in NoC for Ci. Thus, we can obtain a constraint BWimin for initiating the TDM design step independently from the L_i unknown value. This assumption is enforced by the fact that final BW_i values are larger than constraints since the computation is based on the integer TDM slot division. Actually, the real challenge is to find path in order to use the minimum TDM table size. Thus, the method we apply to provide the path allocation algorithm with constraints is based on two steps: 1) a minimum bandwidth is settled by the designer for best effort traffics in each N_I, it means that a minimum of free slots will be available in TDM tables 2) a computation of bandwidth requirements based on Eq.9.

C. TDM table size computation algorithm

1) Mutual exclusive Ci notification.
2) |S| is initialized to 1. It means that the search starts with minimum latency for all Ci.
3) While (Sum of required slots > TDM size & TDM size < max size) ,
   Add a new slot to TDM table,
   Compute number of required slots Si for each Ci
4) All Si and N are available: computation of path length constraints Simax with real bandwidth (based on Si and N) and Eq.9.

The next step is the path allocation based on Simax and Si.

D. Mutual exclusive communications

1) Mutual exclusion (ME) definition and rules: Mutual exclusion improve the probability to find valid paths for a given small TDM table size. We have defined two kinds of ME in the context of NoCs :
- Ci and Cj are strongly exclusive communications if data from Ci (resp. Cj) are entirely consumed before the emission of data over Cj (resp. Ci). At the NoC level, it means that credits are fully setup, i.e. the buffer at destination is empty.
- Ci and Cj are lightly exclusive communications if they can not overlap in time, however it does not guaranty that data from Ci (resp Cj) are consumed entirely before Cj (resp. Ci) starts. It means that they can not use the link at the same time. For both cases, we can share slot reservations of ME GT communications and then obtain a better link utilization leading to a better NoC use. The difference between strong and light ME is observed on buffers size. In the first case the round trip buffer size can be optimally reduced since it is limited to the maximum value among ME communication buffer sizes whereas in the second case optimizations are obtained only if reserved time slots are overlapping as explained hereafter. ME exclusions are specified by the designer in the first step and then automatically organized in cliques in the second design step before TDM sizing. A single communication can belong to several cliques. During path allocation the following rule is applied:
Multiple pre or final reservations can be instantiated for a single slot only if communications are belonging to at least one common clique.

E. Space Time Path Allocation

It's crucial to implement an efficient path allocation algorithm able to find right paths with minimum TDM table size. Our algorithm do choice after pre-reservations phase, to find a optimised solution. This algorithm is detailed in [**?**].

V. Cases Studies and Results

Our derivation technique has been evaluated with different cases studies. Two different applications have been chosen to illustrate both aspects of communication constraint derivation, namely the initialisation and cadence constraints. The first one is an embedded smart camera application implemented object tracking tasks and the second is a multiprocessor turbo-decoder.

1) Context: As explained in introduction, the aim of this study is not the NoC sizing which is decided by the designer with our interactive tool in step 1. NoC specifications are the following: The routing technique is street sign. The router pipeline depth equals 2 and consequently a packet needs two clock cycles to perform a hop from one router to its neighbor.

2) Embedded Smart Camera: This application implements different image processing tasks including video acquisition, image filtering, object and background extraction, object tracking and different display controls. The application architecture is composed of 13 IP Ports: 6 Memories, 6 hardware IPs and 1 Processor. The application required 19 read or

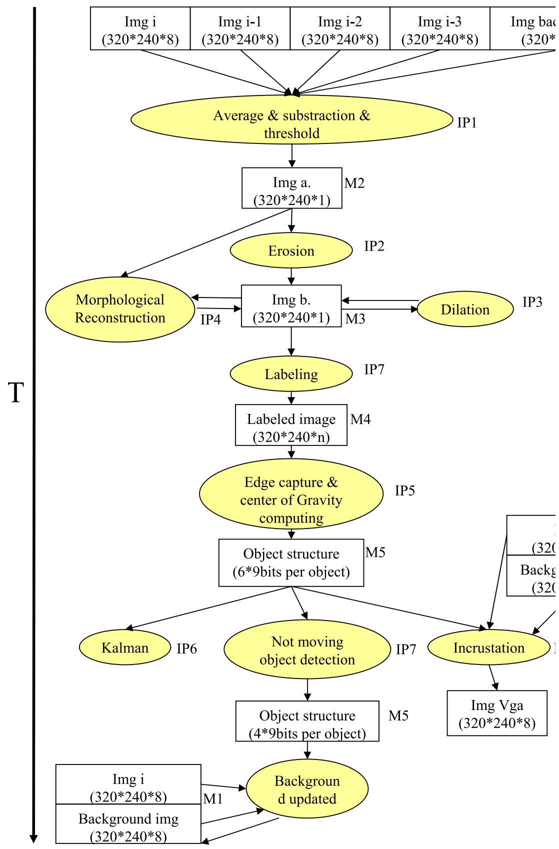

Fig. 5. Embedded Smart Camera Application

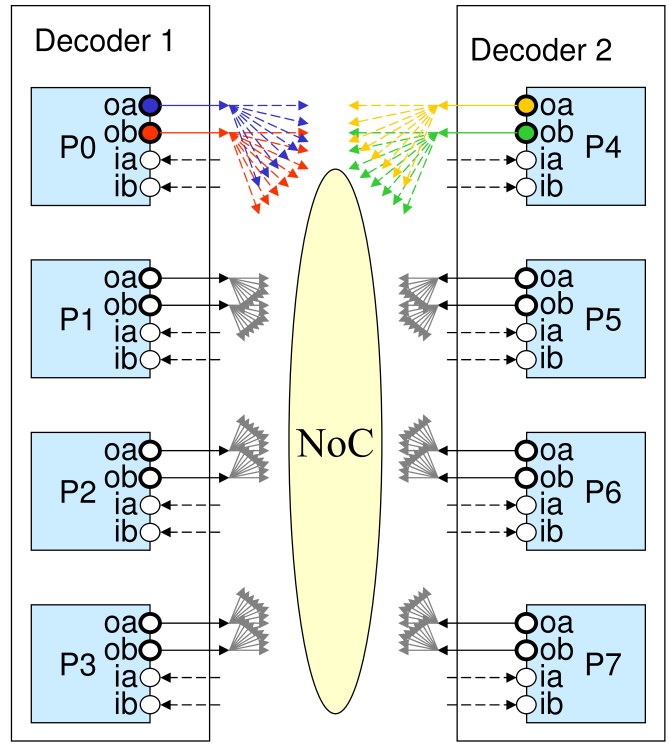

Fig. 6. Turbo decoder communications

write transactions. The image size is 320x240 pixe throughput constraints is 25 frames per second. Figu the algorithm flow.

The smart camera application presents a typical example where initialisation constraints have to be handled. Most of the tasks need a minimum amount of data from their predecessor before starting first iteration. For instance, erosion task requires 2 lines (640 pixels) and 3 pixels before starting the first computation iteration. Another point is the long chain of dependency between tasks, which imply recursive computation of initialising delays according to Eq. 2. This timing verification is mainly based on rule 1 checking.

It is very difficult to provide bandwidth and latency communication constraints assumed known in traditional tools. All those constraints can be specified with our model in our tool. Without such a tool, bandwidth and latency are usually approximated. This leads designers to over constrain the communication specification requirements, with the consequence of a over sizing of the architecture, or unfortunately constraints are wrong evaluated and applications constraints are not met.

3) Multiprocessor Turbo Decoding : This application use the DVB RCS turbo code. Turbo decoder allows to reduce the error rate with a lower signal-to-noise ratio. It can be designed using multiprocessor platform to reach high speed rates [4]. The turbo-decoder is made of two decoder, exchanging information to converge to decisions.

Communication exchanges depend on the shuffling scheme. Our aim is the design of a turbo decoder able to change its shuffling scheme. To carry communications, we chose to use a NoC for its ability to be reconfigured.

In our use case, one extrinsic information is constituted of 72 bits (9 Bytes), moreover each of the two decoders are made of four processors, and each processor has two input and two output channels. Fig.6 shows the architecture of our turbo decoder. Thus the application has 8 processors, and 16 inputs and 16 output ports. Input and output ports of same channel are clustered. Thus, we have 16 NIs, and 128 connections are possible. Only 16 extrinsic informations are produced (one every 0.1 μ seconds). Latency constraint to carry an extrinsic information is 0.30 μs. The shuffling scheme is based on a 16*64 matrix with the following restriction: An input port can't receive more than 5 extrinsic informations during a period of 3 consecutive iterations. This avoids over sizing our communication architecture.

There is not data dependency between tasks. This application is typically cadence oriented. This is the rule 2 which is applied. The difficulty is to take benefits from communication application knowledge to find a solution. Extract communication specifications from the application knowledge is a very tedious and error prone problem. But, without this important specification step, the worst case is considered and lead to a prohibitive architecture cost.

A. Results

1) embedded smart camera application: For the embedded smart camera application, we specify a 3x2 mesh topology, with 5 ports routers, and with 32 bits Phit width (physical units) at 100MHz. Thanks to our approach, during problem solving, appropriately constraints are relaxed when it becomes possible, leading to a cheaper NoC. The 25 frames per second constraint is respected. Slot Table Depth is 5 slots and the sum of buffer depths is 120 words of 32 bits.

2) multiprocessor turbo-decoder application: For the multiprocessor turbo-decoder application, we specify a 4*4 mesh topology, with 4 ports routers, and a frequency of 200MHz. NoC parameters are successfully found out. Slot Table Depth is equal to 10 slots. Buffer cost is 416 words. Path and TDM tables for NI are successfully computed. The largest latency is 0,20 μ seconds, and is so in accordance with the specified constraints.

To achieve application constraints, a 24 bits phit width is sufficient with mutual exclusion when a 72 bits width is needed without mutual exclusion specification. The area cost is so 3 times higher in the second case than in the first one. It is a typical example that shows how a cad tool with formal constraint specifications can help the designers to find a low cost design.

VI. Conclusion

NoC optimal design requires constraint specification for each of its communications. Those communication constraints are bandwidth and latency. Unfortunately, those informations are usually unknown, because communications belong to a complex flow with many interdependencies, and only application level constraints are usually available. A problem like this can only be managed with a formal specification with the definition of mutual exclusion communications and rule checker used during problem solving to appropriately relax constraints when it become possible. We used smart camera and turbo decoder applications to demonstrate our derivation technique. These two real-life applications are especially relevant to prove our approach since the first one mainly constrained by the initialisation phase (checked with rule 1) and the second one is mostly dominated by the cadence constraint (checked with rule 2). Our method is based on a fast rule checker algorithm that provides a solution for taking automatically benefits from system-level application knowledge in order to reduce NoC cost.

References

[1] A. Fanet, "Noc: the arch key of ip integration methodology," in *MPSoC'05 5th International Forum on Application-Specific Multi-Processor SoC*, Relais de Margaux, France, 11 - 15 July 2005.

[2] K. Goossens and J. Dielissen and all., "A design flow for application-specific networks on chip with guaranteed performance to accelerate soc design and verification," in *Proc. Design, Automation and Test in Europe Conference and Exhibition (DATE)*, march 2005.

[3] C. Marcon and M. Kreutz and N. Calazans and A. Susin, "Models for embedded application mapping onto nocs: Timing analysis," vol., 00,

[4] O. Muller and A. Baghdadi and and M. Jezequel, "Asip-based multi-processor soc design for simple and double binary turbo decoding," in *IEEE/ACM Design, Automation and Test in europe*, Munich, Germany, March 6-10 2006.